EXHIBIT 1
PART 1

FILED UNDER SEAL

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
[1.0] A computing device comprising:	Google's "Cast" technology enables an "Android, iOS, or Web app to direct its streaming video and audio to a TV or sound system," where the app "becomes the remote control to play, pause, seek, rewind, stop, and otherwise control the media." https://developers.google.com/cast.[1] In Google's "Cast" framework, there are two core categories of devices: (1) "sender" devices, which are computing devices installed with a Cast-enabled Android, iOS, Chrome, or browser-based app accessed via either an app store or Chromecast-enabled site URL[2] (including youtube.com, music.youtube.com, tv.youtube.com, and spotify.com), and (2) "receiver" devices, which are Cast-enabled media players such as an audio or video playback device. *See, e.g.,* https://developers.google.com/cast/docs/developers; https://developers.google.com/cast/glossary; https://developers.google.com/cast/docs/ux_guidelines. There are many different Cast-enabled Android, iOS, Chrome, or browser-based apps that allow a user to transfer playback of streaming media content from the user's smartphone, tablet, or computer device to a Cast-enabled media player and then control the Cast-enabled media player's playback using the Cast-enabled app. This includes Google's own Cast-enabled apps, such as the YouTube Music app, the YouTube app, the YouTube TV app, and the YouTube Kids app, as well as a host of different third-party Cast-enabled apps, such as the Spotify app. *See, e.g.,* https://support.google.com/chromecastbuiltin/answer/6279384?hl=en#zippy=%2Cbefore-you-begin-casting%2Ccast-from-chromecast-enabled-apps-to-your-audio-device%2Cfind-new-content-to-cast; https://www.google.com/chromecast/built-in/apps/. These Cast-enabled apps can be installed and run on any smartphone, tablet, or computer device that supports Android, iOS, Chrome, or browser-based apps, including Google's own "Pixel" smartphone, tablet, and computer devices (*e.g.*, the Pixel, Pixel XL, Pixel 2, Pixel 2 XL, Pixel 3, Pixel 3 XL, Pixel 3a, Pixel 3a XL, Pixel 4, Pixel 4 XL, Pixel 4a, Pixel 4a (5G), Pixel 5, Pixel 5a (5G), Pixel 6 phones, the Pixel Slate tablet, and the Pixelbook and Pixelbook Go laptops) as well as many third-party smartphone, tablet, or computer device. *See, e.g.,* https://store.google.com/us/magazine/compare_pixel; https://store.google.com/us/product/google_pixelbook_specs; https://store.google.com/us/product/pixel_slate_specs. For purposes of this chart, any smartphone, tablet, or computer device installed with a Cast-enabled Android, iOS, Chrome, or browser-based app (e.g., accessed

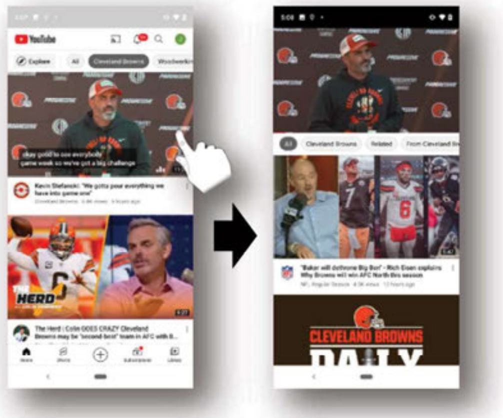
[1] Additional information regarding the accused instrumentalities is set forth in the Infringement Contention Chart for U.S. Patent No. 9,967,615 (Ex. A), which is incorporated herein by reference.
[2] *See, e.g.,* https://support.google.com/chromecast/answer/3265953?hl=en.

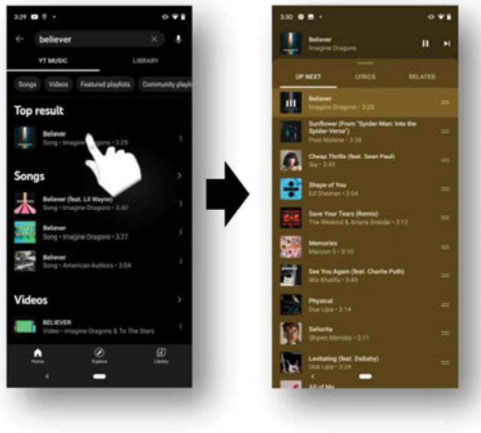
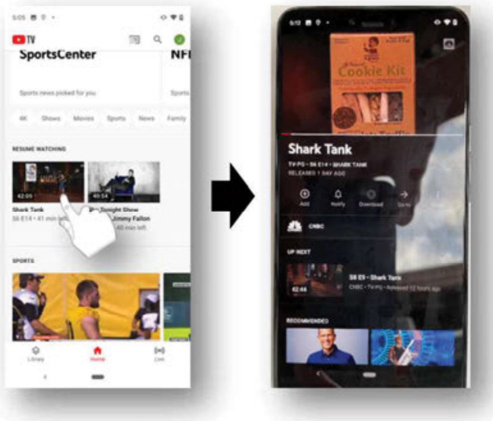
Claim 1	Accused Instrumentalities
	via an app store or Chromecast-enabled site URL) that allows a user to transfer playback of streaming media content from the smartphone, tablet, or computer device to a Cast-enabled media player and then control the Cast-enabled media player's playback using the Cast-enabled app will be referred to as a "Cast-enabled computing device." As set out in the Supplemental Infringement Contentions, Sonos has endeavored to list the possible third-party smartphone, tablet, and computer devices that may have been used to download and install one or more accused Cast-enabled apps between May 2018 to present. *See* Appx. 1. Each of these third-party smartphone, tablet, and computer devices, when installed with at least one Cast-enabled app, meets all elements of the claims as explained in more detail herein. As set forth below, Sonos cites representative examples of Google smartphone, tablet, and computer devices meeting elements 1.1, 1.2, and 1.3, but notes that the hardware aspects recited in elements 1.1, 1.2, and 1.3 have become ubiquitous among all smartphone, tablet, and computer devices. Each possible third-party smartphone, tablet, and computer device listed in Appendix 1 contains these aspects and thus meets elements 1.1, 1.2, and 1.3. There are also many different Cast-enabled media players to which playback of streaming media content can be transferred from a Cast-enabled computing device. This includes Google's own Cast-enabled media players, such as the Home Mini, Nest Mini, Home, Home Max, Home Hub, Nest Hub, Nest Hub Max, Nest Wifi Point, Chromecast, Chromecast Audio, Chromecast Ultra, Chromecast with Google TV, and Nest Audio media players, as well as various other third-party media players with built-in Cast functionality. *See, e.g.,* https://store.google.com/us/product/google_home_max?hl=en-US; https://store.google.com/us/product/chromecast_google_tv_compare?hl=en-US; https://www.google.com/chromecast/built-in/audio/. Certain of these Cast-enabled media players also include a display screen and firmware that enables the Cast-enabled media players to additionally function as a control device for other Cast-enabled media players. This sub-category of Cast-enabled media players, which will be referred to herein as "Cast-enabled displays," includes Google's Home Hub, Nest Hub, and Nest Hub Max media players. *See, e.g.,* https://store.google.com/us/product/google_nest_hub?hl=en-US#overview-modal-music; https://store.google.com/us/product/google_nest_hub_max?hl=en-US; https://support.google.com/googlenest/answer/9165738?hl=en. Similar to the Cast-enabled computing devices, these Cast-enabled displays have Cast-enabled software (e.g., firmware and/or Cast-enabled apps)

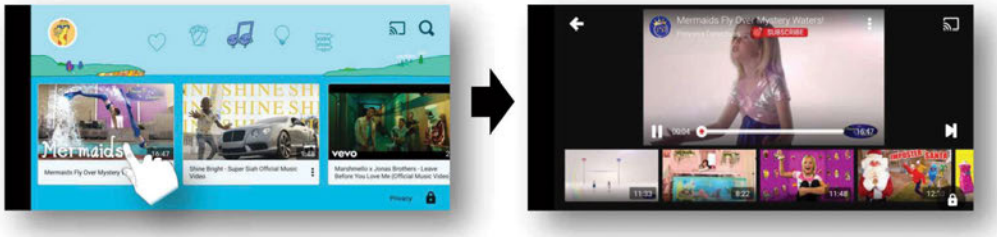
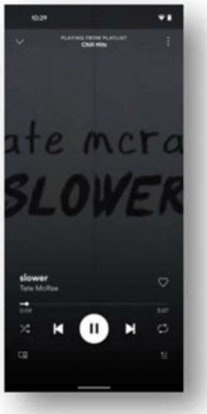
Claim 1	Accused Instrumentalities
	that allows a user to transfer playback of streaming media content from the Cast-enabled display to another Cast-enabled media player and then control that other Cast-enabled media player's playback using the Cast-enabled display's user interface. For purposes of this chart, Cast-enabled computing devices and Cast-enabled displays may be referred to collectively as "Cast-enabled control devices." Google also hosts backend software on Cloud-based infrastructure owned and/or operated by Google (sometimes referred to as Google Cloud Platform or "GCP" for short) that facilitates the aforementioned Cast functionality for transferring playback of streaming media content from a Cast-enabled control device to a Cast-enabled media player and/or controlling the Cast-enabled media player's playback. As described in further detail below, each Cast-enabled control device is a "computing device," as recited in claim 1. Further, because each Cast-enabled media player is a data network device (*i.e.*, a device that is configured to connect to and communicate over a medium that interconnects devices in a manner that enables them to send digital data packets to and receive digital data packets from each other) and is configured to process and output audio, each Cast-enabled media player is a "playback device" as recited in claim 1. *See, e.g.*, Pl.'s Opening Markman Br. (D.I. 60 of 20-cv-881-ADA) at pp. 4-6, 23-26, Exs. 24-25; Pl.'s Reply Markman Br. (D.I. 66 of 20-cv-881-ADA) at pp. 2-3, 10-12, Exs. 26-27; SONOS-SVG2-00018184 - SONOS-SVG2-00018236 [ITC Order No. 20] at p. 15; https://support.google.com/googlenest/answer/7072284?hl=en; https://support.google.com/chromecast/answer/3046409?hl=en; https://store.google.com/us/product/nest_wifi_specs?hl=en-US.
[1.1] at least one processor;	Each Cast-enabled control device includes at least one processor. *See, e.g.,* https://store.google.com/us/magazine/compare_pixel; https://store.google.com/us/product/google_pixelbook_specs; https://store.google.com/us/product/pixel_slate_specs; https://store.google.com/us/product/google_home_max?hl=en-US.
[1.2] a non-transitory computer-readable medium;	Each Cast-enabled control device includes a non-transitory computer-readable medium. *See, e.g.,* https://store.google.com/us/magazine/compare_pixel; https://store.google.com/us/product/google_pixelbook_specs; https://store.google.com/us/product/pixel_slate_specs; https://store.google.com/us/product/google_home_max?hl=en-US.

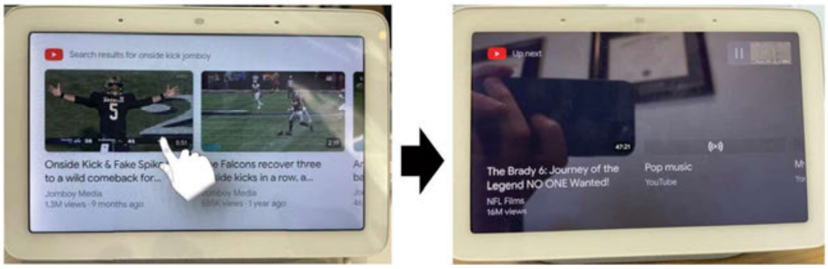
Claim 1	Accused Instrumentalities
[1.3] and program instructions stored on the non-transitory computer-readable medium that, when executed by the at least one processor, cause the computing device to perform functions comprising:	Each Cast-enabled control device includes program instructions stored on the non-transitory computer-readable medium that enable the Cast-enabled control device to perform the functions identified below. *See, e.g.,* https://store.google.com/us/magazine/compare_pixel; https://store.google.com/us/product/google_pixelbook_specs; https://store.google.com/us/product/pixel_slate_specs; https://store.google.com/us/product/google_home_max?hl=en-US.
[1.4] operating in a first mode in which the computing device is configured for playback of a remote playback queue provided by a cloud-based computing system associated with a cloud-based media service;	Each Cast-enabled control device comprises program instructions stored on the Cast-enabled control device's non-transitory computer-readable medium that, when executed by the Cast-enabled control device's processor, cause the Cast-enabled control device to operate in a first mode in which the Cast-enabled control device is configured for playback of a remote playback queue provided by a cloud-based computing system associated with a cloud-based media service. For instance, each Cast-enabled computing device is programmed with the capability to operate in a mode in which the Cast-enabled computing device is configured for playback of a remote playback queue provided by a cloud-based computing system that takes the form of one or more cloud servers associated with a cloud-based media service (e.g., a Google service such as YouTube, YouTube Music, YouTube TV, YouTube Kids, etc. or a third-party service such as Spotify, etc.) that are remote from the Cast-enabled computing device and accessible over the Internet, which may be operated by Google or a third-party service provider. *See, e.g.,* https://support.google.com/googlenest/answer/7181830 [Play media from Chromecast-enabled apps to your speaker or display]; https://support.google.com/chromecast/answer/2995235?hl=en-AU [Cast from the YouTube app and YouTube.com]; https://support.google.com/googlenest/answer/9563059?hl=en-IN [Move media from one cast device to another]; https://support.google.com/youtubetv/answer/7353493?co=GENIE.Platform%3DAndroid&hl=en [Cast YouTube TV using Chromecast]; https://support.google.com/youtubekids/answer/6289408?hl=en&co=GENIE.Platform%3DAndroid [Watch

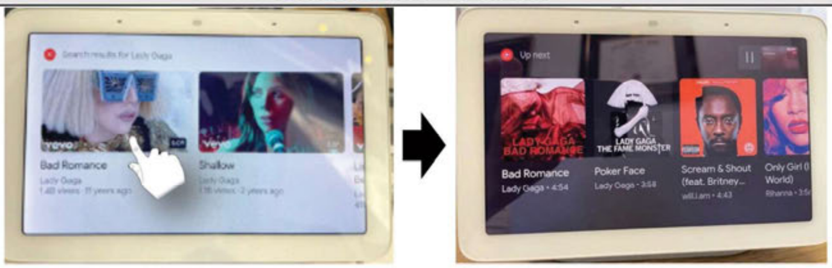
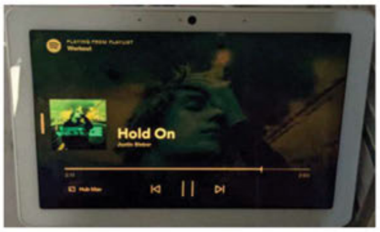
Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	YouTube Kids videos on your TV]; https://support.google.com/chromecast/answer/3265953?hl=en [Chromecast-enabled site vs. casting a tab]; https://support.google.com/youtube/answer/6327615?hl=en [Autoplay videos]; https://developers.google.com/cast/docs/web_receiver/queueing; https://developers.google.com/cast/docs/ios_sender/queueing; https://developers.google.com/cast/docs/android_sender/queueing; *see also, e.g.*, GOOG-SONOSWDTX-00042745 [Orbit Queue - Android Implementation 11.07.2017] at 46 ("The queue is overall seen as a list of videos that the user plans to see in this session."); GOOG-SONOSWDTX-00037841 [Episode Queue design doc 07.15.2019] at 41 ("An episode queue provides users with a tool to select what episodes they want to play next . . . ."). Cast-enabled computing devices installed with various of Google's own Cast-enabled apps are programmed to perform this functionality, including but not limited to the YouTube, YouTube Music, YouTube TV, and YouTube Kids apps, as illustrated by the following screenshots:  YouTube Explore All Cleveland Browns Kevin Stefanski: "We gotta pour everything we have into game one" HERD The Herd | Colin GOES CRAZY Cleveland Browns may be "second-best" team in AFC with B... All Cleveland Browns Related From Cleveland Br "Baker will dethrone Big Ben" - Rich Eisen explains Why Browns will win AFC North this season CLEVELAND BROWNS DAILY

Claim 1	Accused Instrumentalities
	 3:29 believer YT MUSIC LIBRARY Songs Videos Featured playlists Community playl Top result Believer Song • Imagine Dragons • 3:25 Songs Believer (feat. Lil Wayne) Song • Imagine Dragons • 3:40 Believer Song • Imagine Dragons • 3:27 Believer Song • American Authors • 3:04 Videos BELIEVER Video • Imagine Dragons & To The Stars Home Explore Library 3:30 Believer Imagine Dragons UP NEXT LYRICS RELATED Believer Imagine Dragons • 3:25 Sunflower (From "Spider-Man: Into the Spider-Verse") Post Malone • 2:38 Cheap Thrills (feat. Sean Paul) Sia • 3:45 Shape of You Ed Sheeran • 3:54 Save Your Tears (Remix) The Weeknd & Ariana Grande • 3:12 Memories Maroon 5 • 3:10 See You Again (feat. Charlie Puth) Wiz Khalifa • 3:49 Physical Dua Lipa • 3:14 Señorita Shawn Mendes • 3:11 Levitating (feat. DaBaby) Dua Lipa • 3:24 All of Me  5:05 TV SportsCenter NFI Sports news picked for you Sports 4K Shows Movies Sports News Family RESUME WATCHING 42:09 40:54 Shark Tank S6 E14 • 41 min left The Tonight Show Jimmy Fallon 40 min left SPORTS Library Home Live 5:12 Cookie Kit Shark Tank TV-PG • S6 E14 • SHARK TANK RELEASED 1 DAY AGO Add Notify Download Go to CNBC UP NEXT 42:44 S6 E9 • Shark Tank CNBC • TV-PG • Released 12 hours ago RECOMMENDED

Claim 1	Accused Instrumentalities
	 Mermaids vevo Mermaids Fly Over Mystery Waters! SUBSCRIBE Likewise, Cast-enabled computing devices installed with various third-party Cast-enabled apps are also programmed to perform this functionality, including but not limited to the Spotify app, as illustrated by the following screenshot:  PLAYING FROM PLAYLIST Chill Hits ate mcra SLOWER slower Tate McRae Various other Cast-enabled apps available for installation on Cast-enabled computing devices provide similar functionality. *See, e.g.*, https://support.google.com/chromecast/answer/6279384?hl=en [Cast audio from Chromecast-enabled apps to speakers]; https://www.google.com/chromecast/built-in/apps/.

Claim 1	Accused Instrumentalities
	Each Cast-enabled display is also programmed with the capability to operate in a mode in which the Cast-enabled display is configured for playback of a remote playback queue provided by a cloud-based computing system that takes the form of one or more cloud servers associated with a cloud-based media service (e.g., a Google service such as YouTube, YouTube Music, etc., or a third-party service such as Spotify, etc.) that are remote from the Cast-enabled display and accessible over the Internet, which may be operated by Google or a third-party service provider. *See, e.g.*, https://store.google.com/us/product/google_nest_hub?hl=en-US#overview-modal-music ("YouTube Music on demand. . . . Stream top music services."); https://store.google.com/us/product/google_nest_hub_max?hl=en-US ("[J]am out with YouTube Music."); https://support.google.com/googlenest/answer/9165738?hl=en ("With YouTube built-in to your Google Nest display, you can watch YouTube Originals, how-to videos and much more, seamlessly on your screen."). Cast-enabled displays installed with various of Google's own Cast-enabled apps are programmed to perform this functionality, including but not limited to the YouTube and YouTube Music apps, as illustrated by the following photos:  Search results for onside kick jomboy Onside Kick & Fake Spik to a wild comeback for... Jomboy Media 1.3M views · 9 months ago e Falcons recover three ide kicks in a row, a... omboy Media views · 1 year ago Up next 47:21 The Brady 6: Journey of the Legend NO ONE Wanted! NFL Films 16M views Pop music YouTube

Claim 1	Accused Instrumentalities
	 Search results for Lady Gaga Bad Romance Shallow Up next Bad Romance Poker Face Scream & Shout (feat. Britney... Only Girl (In the World) Additionally, Cast-enabled displays installed with various third-party Cast-enabled apps are also programmed to perform this functionality, including but not limited to the Spotify app, as illustrated by the following photo:  Various other Cast-enabled apps available for installation on Cast-enabled displays provide similar functionality. Further details regarding how a Cast-enabled control device carries out this functionality, along with representative examples of other evidence demonstrating this functionality, are summarized below. ***<u>YouTube, YouTube Music, YouTube TV, & YouTube Kids apps</u>*** Each Cast-enabled computing device installed with any one of the YouTube, YouTube Music, YouTube TV, or YouTube Kids app is programmed such that it can operate in a mode in which the Cast-enabled computing

Claim 1	Accused Instrumentalities
	device is configured for playback of a remote playback queue (referred to herein as a "Watch Next" queue) provided by one or more cloud servers (e.g., a "Watch Next," "InnerTube," or "MDx" server) associated with the YouTube, YouTube Music, YouTube TV, or YouTube Kids media service. The aforementioned functionality satisfies claim limitation 1.4. Faced with this clear evidence that Google's Cast-enabled computing devices are capable of playing back a "remote playback queue" as required by claim limitation 1.4, and that Google's Cast-enabled computing devices and Cast-enabled media players also utilize such a "remote playback queue" in the manner recited by the remaining limitations of claim 1 (as explained in further detail below), Google has now proposed a brand new construction for the term "playback queue," which Google appears to be pursuing for the sole purpose of attempting to avoid infringement of the claim limitations that reference a "remote playback queue." In particular, for the first time on February 3, 2022, Google took the position that the term "playback queue" should be construed as "[a]n ordered list of multimedia items that is selected by the user for playback." In this regard, Google apparently intends to argue that the accused "Watch Next" queue is not a "remote playback queue" because it is not "[a]n ordered list of multimedia items that is selected by the user for playback," although Google has not provided any explanation as to exactly why it has introduced this construction of "playback queue," and it is still not clear how Google intends to interpret or apply this construction of "playback queue" in the context of the claims of the '033 Patent. In any event, Sonos disagrees that this is the proper construction for "playback queue" as that term is used in the context of the '033 Patent and will provide its position regarding the flaws in Google's proposed construction during the claim construction process. However, even if the Court were to adopt such a construction for "playback queue," Sonos maintains that the accused "Watch Next" queue would still amount to the claimed "remote playback queue" either literally or at the very least under the Doctrine of Equivalents ("DoE"), and because of this, each Cast-enabled computing device installed with any one of the YouTube, YouTube Music, YouTube TV, or YouTube Kids apps would still satisfy the claim limitations that reference a "remote playback queue" either literally or at the very least under DoE. As established by the evidence cited herein, when a user initiates local playback of user-selected media content from a YouTube, YouTube Music, YouTube TV, or YouTube Kids service on a Cast-enabled computing device, this causes the Cast-enabled computing device to become configured for playback of a playback queue referred to herein as a "Watch Next" queue that is remote from the Cast-enabled computing

Claim 1	Accused Instrumentalities
	device and will contain (i) a locator of at least one media item that was selected by the user for playback along with (ii) locators for additional media items that were identified by the applicable YouTube service based on the user's selection of the media content and are seeded for playback after the user-selected media content, which are referred to as "Autoplay"[3] media items. This is evidenced by the data object called playlist_util.py→PlaylistFiller.playlist_videos. *See also, e.g.*, GOOG-SONOSWDTX-00052111 [YouTube Developer's Handbook – Life of a video recommendation] at 11-12 ("Every time a user loads the YouTube homepage or a video watch page, YouTube suggests other videos to watch. . . . Recommendations, or recs, are suggestions based on what YouTube might already know about the user, through the user's past interactions on the YouTube platform and their current activity. . . . When a user loads a video, clicks on a recommendation from the Home page, or clicks on a search result, they see a video watch page containing their chosen video, along with a queue of other recommendations under Up next."); GOOG-SONOSWDTX-00039673 [YTM Playback Squad – Life of a playback 12.22.2020] at 73-74 ("Queues are sourced from a WatchNext response. . . . The first WatchNext response in a container always contains the queue. . . . Always use InnerTubeUtil to retrieve the queue . . . . The queue is represented by a PlaylistPanelRenderer where each queue item is a PlaylistPanelVideoRenderer or a PlaylistPanelVideoWrapperRenderer . . . . They both implement VideoItem which is the base type for a queue item."). As noted above, the first aspect of Google's new construction of "playback queue" is that it requires an "ordered list of multimedia items." The accused "Watch Next" queue clearly meets this "ordered list of multimedia items" aspect of Google's new construction, at least by virtue of the fact that the "Watch Next" queue is configured to contain locators of media items for playback in a sequential order. *See, e.g.*, GOOG-SONOSWDTX-00052111 [YouTube Developer's Handbook – Life of a video recommendation]; GOOG-SONOSWDTX-00039673 [YTM Playback Squad – Life of a playback 12.22.2020]; GOOG-SONOSWDTX-00039785 [YTM Playback Squad – Server 01.05.2021]. Sonos does not see any credible basis for Google to contend otherwise, but it is also not clear how Google intends to interpret or apply the phrase "ordered list" in

[3] https://support.google.com/youtube/answer/6327615?hl=en [Autoplay videos]; https://support.google.com/youtubekids/answer/6130531?hl=en [Recommended videos]; https://support.google.com/youtubekids/answer/6138623?hl=en&co=GENIE.Platform%3DAndroid [Accessibility on YouTube Kids] ("When autoplay is turned on, we'll automatically play another related video."); GOOG-SONOSWDTX-00005974 [YouTube Help – Autoplay videos]; GOOG-SONOSWDTX-00039798 [YTM Playback Squad – The Queue 09.30.2020] at 99 ("The GetMusicWatchNext endpoint provides queue renderers when initiating playback on a new container, on queue continuation loads, and for the autoplay section of the queue.").

Claim 1	Accused Instrumentalities
	the context of its new construction, and Sonos expressly reserves its right to further supplement its infringement contentions if Google later attempts to advance a new interpretation of the phrase "ordered list." Turning to the second aspect of Google's new construction of "playback queue," Google is also now attempting to add a requirement that the "ordered list of multimedia items" be "selected by the user for playback." Notably, Google has yet to provide any basis for its position that this new limitation is a required aspect of a "playback queue," and it is still not clear how Google intends to interpret or apply this new limitation in the context of its construction. This is particularly the case given that Google seems to be defining the "remote playback queue" – which is a data structure that is configured to contain an identification of whatever media content is queued for playback at a given time (*i.e.*, a container) – in terms of the unrelated details as to how the media items contained within the "remote playback queue" were previously selected, which would result in a nonsensical interpretation of the claims where a data structure would qualify as a "remote playback queue" during some periods of time (*i.e.*, when it contains user-selected media items) and would not qualify as a "remote playback queue" during other periods of time (*i.e.*, when it does not contain user-selected media items) despite the fact that it is the exact same data structure and is being used in the exact same manner to facilitate playback. Nevertheless, Google appears to have imported this new limitation into its construction so that it can then argue that, because the accused "Watch Next" queue may contain some media items that were directly selected by a user and other items that were not directly selected by a user (e.g., media items identified by a YouTube service based on the user's selection), the accused "Watch Next" queue is not a "remote playback queue." However, even setting aside the flaws in Google's construction (which will be addressed during the claim construction process), such a non-infringement argument fails for several reasons. ***First***, the evidence cited herein establishes that the "Watch Next" queue will contain a locator of at least an initial media item that was directly selected by the user for playback along with locators of additional "Autoplay" media items that were identified based on the user's selection of the initial media item. In this respect, the "Watch Next" queue literally amounts to an "ordered list of multimedia items that is selected by the user for playback" because the initial media item in the "Watch Next" queue was selected by the user and the additional media items were then identified based on the user's selection. Thus, because each Cast-enabled computing device installed with any one of the YouTube, YouTube Music, YouTube TV, or YouTube Kids apps is programmed to perform the functionality of claim limitation 1.4 (as well as the other claim limitations that refer to a "remote playback queue"), with respect to the "Watch Next" queue that

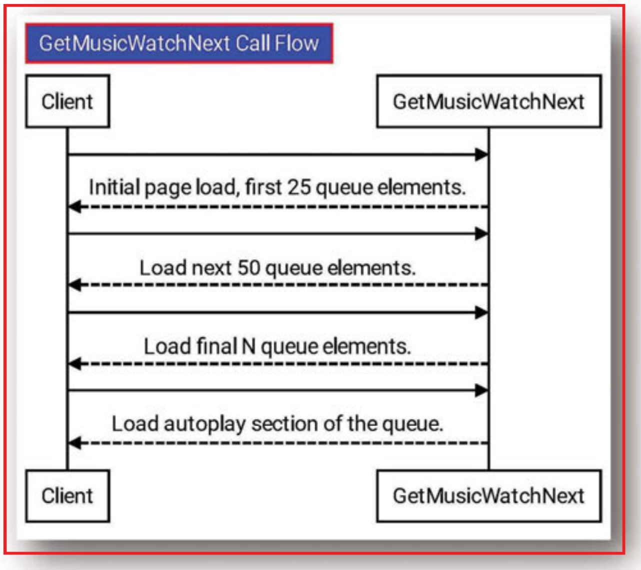
Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	literally amounts to "an ordered list of multimedia items that is selected by the user for playback," each such Cast-enabled computing device would still literally satisfy claim limitation 1.4 (as well as the other claim limitations that refer to a "remote playback queue") under Google's construction for this additional reason. ***Second***, to the extent that the Court adopts Google's construction of "playback queue" and Google then later tries to argue that a the "Watch Next" queue does not literally amount to an "ordered list of multimedia items that is selected by the user for playback" unless every single media item in the "Watch Next" queue is directly selected by the user, the relevant functionality carried out by a Cast-enabled computing device with respect to a "Watch Next" queue containing at least one media item that was directly selected by a user and other items that were not directly selected by a user (e.g., media items identified by a YouTube service based on the user's selection) still satisfies claim limitation 1.4 (as well as the other claim limitations that refer to a "remote playback queue") under DoE. This is because there is merely an insubstantial difference between (i) a Cast-enabled computing device (or Cast-enabled media player) having responsibility for playback of a "remote playback queue" in which all of the media items were directly selected by a user and (ii) a Cast-enabled computing device (or Cast-enabled media player having responsibility) for playback of a "remote playback queue" in which only the initial media item was directly selected by the user while the other media item was identified based on the user's selection of the initial media item. Indeed, a Cast-enabled computing device performs the same function (e.g., operating in a first mode in which it is configured for playback of the "Watch Next" queue), in the same way (e.g., by interacting with the one or more cloud servers providing the "Watch Next" queue), to achieve the same result (e.g., playing back media items from the "Watch Next" queue) ***regardless*** of whether the media items in the "Watch Next" queue were all directly selected by a user or only the initial media item in the "Watch Next" queue was directly selected by the user and the rest were identified based on the user's selection. And likewise, the cloud-based computing system that provides the "Watch Next" queue performs the same function (e.g., maintaining a "remote playback queue"), in the same way (e.g., by storing locators of media items for playback), to achieve the same result (e.g., providing a "remote playback queue" for playback by a Cast-enabled computing device) ***regardless*** of whether the media items in the "Watch Next" queue were all directly selected by a user or only the initial media item in the "Watch Next" queue was directly selected by the user and the rest were identified based on the user's selection. For all of the foregoing reasons, Sonos maintains that, even if the Court were to adopt Google's new construction for "playback queue," each Cast-enabled computing device installed with any one of the

Claim 1	Accused Instrumentalities
	YouTube, YouTube Music, YouTube TV, or YouTube Kids apps would still satisfy claim limitation 1.4 (as well as the other claim limitations that reference the "remote playback queue") literally, or at the very least, under DoE. While Sonos has made its best effort to interpret and understand Google's evolving construction of "remote playback queue" / "playback queue," and to provide Sonos's infringement position under that evolving construction, it remains unclear how Google intends to interpret and apply that construction to the accused instrumentalities. As such, Sonos expressly reserves the right to further supplement its infringement contentions if Google later attempts to advance an interpretation of this construction that differs from Sonos's current understanding. The following exemplary evidence demonstrates that each Cast-enabled control device installed with the YouTube, YouTube Music, YouTube TV, or YouTube Kids app is programmed with this functionality: • GOOG-SONOSWDTX-00005974 [YouTube Help – Autoplay videos]; • https://support.google.com/youtubekids/answer/6130531?hl=en [Recommended videos]; • https://support.google.com/youtubekids/answer/6138623?hl=en&co=GENIE.Platform%3DAndroid [Accessibility on YouTube Kids] ("When autoplay is turned on, we'll automatically play another related video."); • GOOG-SONOSWDTX-00052111 [YouTube Developer's Handbook – Life of a video recommendation] at 11-12 ("Every time a user loads the YouTube homepage or a video watch page, YouTube suggests other videos to watch. . . . When a user loads a video, clicks on a recommendation from the Home page, or clicks on a search result, they see a video watch page containing their chosen video, along with a queue of other recommendations under Up next."), at 13 ("Whenever a YouTube user visits the Home page or a watch page, a Recs Server receives a request for a list of videos the user might want to watch next, based on factors such as the user's watch history and their other interactions within the YouTube app. The Recs server also referred to as the *suggestion* server for Watch Next recommendations."); • GOOG-SONOSWDTX-00052121 [YouTube Developer's Handbook – Life of a video watch 01.11.2021] at 23-24, 27; • GOOG-SONOSWDTX-00041467 [WatchNext – Watchnext Organizer 01.15.2021];

Claim 1	Accused Instrumentalities
	• GOOG-SONOSWDTX-00039673 [YTM Playback Squad – Life of a playback 12.22.2020] at 73-74 ("Queues are sourced from a WatchNext response. . . . The first WatchNext response in a container always contains the queue. . . . Always use InnerTubeUtil to retrieve the queue . . . . The queue is represented by a PlaylistPanelRenderer where each queue item is a PlaylistPanelVideoRenderer or a PlaylistPanelVideoWrapperRenderer . . . . They both implement VideoItem which is the base type for a queue item."); • GOOG-SONOSWDTX-00039778 [YTM Playback Squad – Modular Player Page 09.17.2020] ("The queue items are returned in the first (as of Sept 2020) TabRenderer's contents, within a newly created MusicQueueRenderer."); • GOOG-SONOSWDTX-00039785 [YTM Playback Squad – Server 01.05.2021] at 87-88 ("The PlaylistDocumentService provides a representation of the queue. It calls into the PlaylistService for video IDs in the playlist . . . . The content set in inbound GetMusicWatchNext requests determines which RPCs are fired and what data is returned to clients. For example, requests for the next track in the queue do not require returning the full queue, and as such make different RPCs and are significantly faster than initial playbacks requiring the queue."), at 88;

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	 GetMusicWatchNext Call Flow Client GetMusicWatchNext Initial page load, first 25 queue elements. Load next 50 queue elements. Load final N queue elements. Load autoplay section of the queue. Client GetMusicWatchNext - *Id.* at 90-91; - GOOG-SONOSWDTX-00039798 [YTM Playback Squad – The Queue 09.30.2020] at 99 ("The GetMusicWatchNext endpoint provides queue renderers when initiating playback on a new container, on queue continuation loads, and for the autoplay section of the queue."). Representative excerpts of Google's YouTube app source code[4] related to the aforementioned functionality include: /java/com/google/android/libraries/youtube/player/internal/queue/InternalQueueController.java

[4] Root directory: /2020-09-22-youtube_android_15.38.35/

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	/java/com/google/android/libraries/youtube/player/service/PlaybackService.java /java/com/google/android/libraries/youtube/player/net/PlayerRequestManager.java /java/com/google/android/libraries/youtube/player/net/OnlinePlaybackRequester.java /java/com/google/android/libraries/youtube/player/net/WatchNextFetcher.java /java/com/google/android/libraries/youtube/innertube/services/watchnext/WatchNextService.java /java/com/google/android/libraries/youtube/innertube/services/InnertubeService.java /java/com/google/android/libraries/youtube/player/service/RequestFlowListener.java /java/com/google/android/libraries/youtube/player/sequencer/OmegaSequencer.java /java/com/google/android/libraries/youtube/player/sequencer/navigation/AutoplaySetSequenceNavigator.java [redacted]

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	[redacted] Representative excerpts of Google's YouTube Music app source code[5] related to the aforementioned functionality include: /java/com/google/android/libraries/youtube/player/internal/queue/InternalQueueController.java /java/com/google/android/libraries/youtube/player/service/PlaybackService.java /java/com/google/android/libraries/youtube/player/net/PlayerRequestManager.java /java/com/google/android/libraries/youtube/player/net/OnlinePlaybackRequester.java /java/com/google/android/libraries/youtube/player/net/WatchNextFetcher.java /java/com/google/android/libraries/youtube/innertube/services/watchnext/WatchNextService.java /java/com/google/android/libraries/youtube/innertube/services/InnertubeService.java /java/com/google/android/libraries/youtube/player/service/RequestFlowListener.java /java/com/google/android/libraries/youtube/player/sequencer/OmegaSequencer.java /java/com/google/android/libraries/youtube/player/sequencer/navigation/AutoplaySetSequenceNavigator.java

[5] Root directory: /2020-10-08-youtube-music_3.87.53/

Claim 1	Accused Instrumentalities
	[redacted]

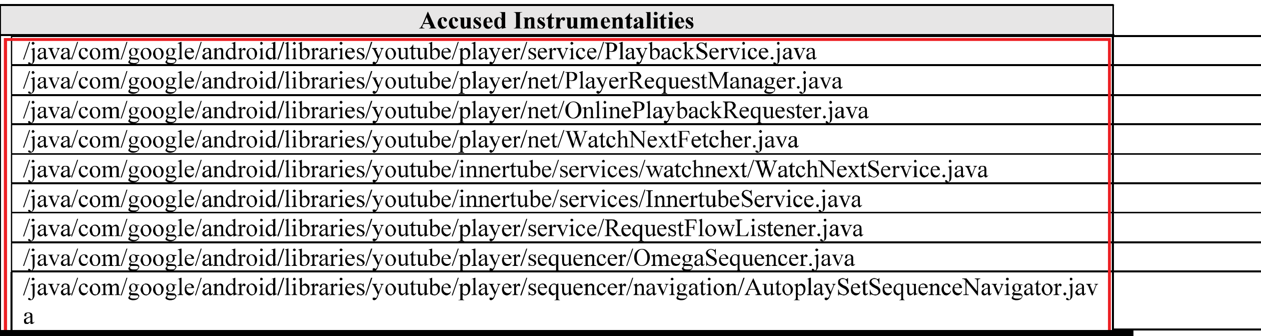
Claim 1	Accused Instrumentalities
	[redacted] Representative excerpts of Google's YouTube TV app source code[6] related to the aforementioned functionality include: /java/com/google/android/libraries/youtube/player/internal/queue/InternalQueueController.java /java/com/google/android/libraries/youtube/player/service/PlaybackService.java /java/com/google/android/libraries/youtube/player/net/PlayerRequestManager.java /java/com/google/android/libraries/youtube/player/net/OnlinePlaybackRequester.java /java/com/google/android/libraries/youtube/player/net/WatchNextFetcher.java /java/com/google/android/libraries/youtube/innertube/services/watchnext/WatchNextService.java /java/com/google/android/libraries/youtube/innertube/services/InnertubeService.java /java/com/google/android/libraries/youtube/player/service/RequestFlowListener.java /java/com/google/android/libraries/youtube/player/sequencer/OmegaSequencer.java /java/com/google/android/libraries/youtube/player/sequencer/navigation/AutoplaySetSequenceNavigator.java [redacted]

[6] Root directory: /2020-09-29-youtube_tv_4.38.3/

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	[redacted] Representative excerpts of Google's YouTube Kids app source code[7] related to the aforementioned functionality include: /java/com/google/android/libraries/youtube/player/internal/queue/InternalQueueController.java

[7] Root directory: /2020-09-28-youtube_kids_5.43.3/

Claim 1	Accused Instrumentalities
	 /java/com/google/android/libraries/youtube/player/service/PlaybackService.java /java/com/google/android/libraries/youtube/player/net/PlayerRequestManager.java /java/com/google/android/libraries/youtube/player/net/OnlinePlaybackRequester.java /java/com/google/android/libraries/youtube/player/net/WatchNextFetcher.java /java/com/google/android/libraries/youtube/innertube/services/watchnext/WatchNextService.java /java/com/google/android/libraries/youtube/innertube/services/InnertubeService.java /java/com/google/android/libraries/youtube/player/service/RequestFlowListener.java /java/com/google/android/libraries/youtube/player/sequencer/OmegaSequencer.java /java/com/google/android/libraries/youtube/player/sequencer/navigation/AutoplaySetSequenceNavigator.jav a [redacted]

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	[redacted] Representative excerpts of Google's server source code[8] related to the aforementioned functionality include: /google3/video/youtube/api/innertube/proto/innertube_service.proto /google3/video/youtube/api/innertube/proto/watch_next/services/innertube_watch_next_service.proto /google3/video/youtube/src/python/servers/innertube/watch_next/innertube_watch_next.py /google3/video/youtube/src/python/servers/innertube/watch_next/_content.py /google3/video/youtube/src/python/servers/innertube/watch_next/_navigation_list.py /google3/video/youtube/src/python/servers/innertube/watch_next/wn_rpc_manager.py /google3/video/youtube/src/python/servers/innertube/watch_next/_playlist_rpc_container.py /google3/video/youtube/src/python/servers/innertube/watch_next/_rpc_manager.py /google3/video/youtube/src/python/servers/innertube/watch_next/_watch_next_rpcs.py /google3/video/youtube/src/python/servers/innertube/watch_next/util.py /google3/video/youtube/src/python/servers/innertube/watch_next/playlist_util.py [redacted]

[8] Root directory: /2021-02-02_YTServerInnerTubeWatchNext09292020/

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	[redacted]

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

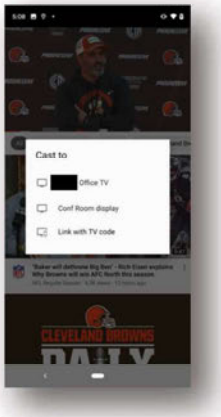
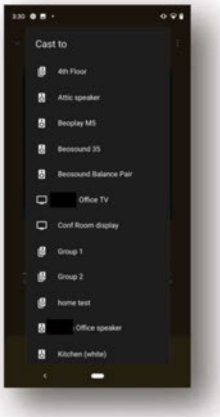
Claim 1	Accused Instrumentalities
	[redacted] Sonos further incorporates by reference Google's response to Sonos's Fact Discovery Interrogatory No. 15, including any of Google's documents or source code cited therein. *See* Google LLC's First Objections and Responses to Plaintiff Sonos, Inc.'s First Set of Fact Discovery Interrogatories. ***Spotify app*** Each Cast-enabled computing device installed with the Spotify app is programmed such that it can operate in a mode in which the Cast-enabled computing device is configured for playback of a remote playback queue (e.g., a "Spotify queue") provided by one or more cloud servers associated with the Spotify media service. The aforementioned functionality satisfies claim limitation 1.4.

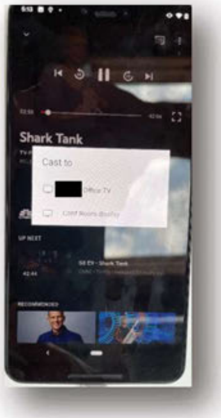
Claim 1	Accused Instrumentalities
	Sonos further incorporates by reference Google's response to Sonos's Fact Discovery Interrogatory No. 15, including any of Google's documents or source code cited therein. *See* Google LLC's First Objections and Responses to Plaintiff Sonos, Inc.'s First Set of Fact Discovery Interrogatories. ***Cast-Enabled Displays*** Each Cast-enabled display is programmed such that it can operate in a mode in which the Cast-enabled display is configured for playback of a remote playback queue provided by one or more cloud servers associated with a particular media service that the Cast-enabled display was selected to playback from (e.g., YouTube, YouTube Music, Spotify, etc.). The aforementioned functionality satisfies claim limitation 1.4. The following exemplary evidence demonstrates that each Cast-enabled display is programmed with this functionality: • https://developers.google.com/cast/docs/web_receiver/queueing?hl=en ("Queueing allows partner applications to better integrate with Cast by providing the following features: Support of Google's and partner's cloud queue implementation so externally stored and created queue can be directly loaded into Cast devices."). Representative excerpts of Google's Cast-enabled display source code[9] related to the aforementioned functionality include: /assistant/display/cast/media/media_session_manager.ts /chrome/dongle/receiver/js/medianamespace.js /chrome/dongle/receiver/js/mediamanager.js /chrome/dongle/receiver/js/mediaqueue.js [redacted]

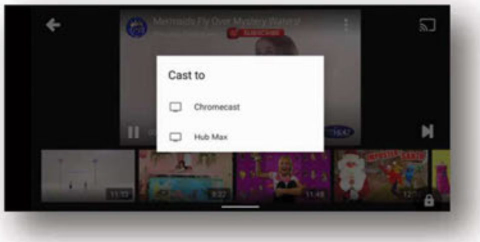
[9] Root directory: /2020-09-01-google3/

Claim 1	Accused Instrumentalities
	[redacted] Sonos further incorporates by reference Google's response to Sonos's Fact Discovery Interrogatory No. 15, including any of Google's documents or source code cited therein. *See* Google LLC's First Objections and Responses to Plaintiff Sonos, Inc.'s First Set of Fact Discovery Interrogatories.
[1.5] while operating in the first mode, displaying a representation of one or more playback devices in a media playback system that are each i) communicatively coupled to the computing device over a data network and ii) available to accept playback responsibility for	Each Cast-enabled control device comprises program instructions stored on the Cast-enabled control device's non-transitory computer-readable medium that, when executed by the Cast-enabled control device's processor, cause the Cast-enabled control device to, while operating in the first mode, display a representation of one or more Cast-enabled media players in a Cast-enabled playback system that are each (i) communicatively coupled to the Cast-enabled control device over a data network and (ii) available to accept playback responsibility for the remote playback queue. For instance, each Cast-enabled computing device is programmed such that, while operating in a mode in which the Cast-enabled computing device is configured for playback of a remote playback queue provided by a cloud server associated with a cloud-based media service (e.g., a Google service such as YouTube, YouTube Music, YouTube TV, YouTube Kids, etc., or a third-party service such as Spotify, etc.), the Cast-enabled computing device is operable to detect a selection of a displayed selectable option (e.g., a selectable "Cast button") for transferring playback of audio content (e.g., music, podcasts, etc.) and/or audiovisual content (e.g., videos) from the Cast-enabled computing device to another device, which triggers the Cast-enabled computing device to display a list of available devices for transferring playback that includes one or more Cast-enabled media players in a Cast-enabled playback system that are each (i) communicatively coupled to the Cast-enabled computing device over a Wi-Fi network and (ii) available to accept playback

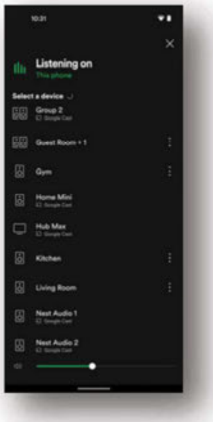
Claim 1	Accused Instrumentalities
the remote playback queue;	responsibility for the remote playback queue. *See, e.g.,* https://support.google.com/googlenest/answer/7181830 [Play media from Chromecast-enabled apps to your speaker or display] ("Tap the Cast button . . . . Tap the speaker or display for which you'd like to cast."); https://support.google.com/chromecast/answer/6279384?hl=en [Cast audio from Chromecast-enabled apps to speakers] ("2. In the top right corner, tap the Cast button . 3. Choose your speaker."); https://support.google.com/chromecast/answer/2995235?hl=en-AU [Cast from the YouTube app and YouTube.com] ("Tap the Cast button . . . . Tap the Chromecast device to which you want to cast."); https://support.google.com/youtubetv/answer/7353493?co=GENIE.Platform%3DAndroid&hl=en [Cast YouTube TV using Chromecast] ("Tap Cast . This is found at the top of the app Home screen. [ ]Choose the device you want to cast to."); https://support.google.com/youtubekids/answer/6289408?hl=en&co=GENIE.Platform%3DAndroid [Watch YouTube Kids videos on your TV]; https://support.google.com/chromecast/answer/3265953?hl=en [Chromecast-enabled site vs. casting a tab]. Under the plain and ordinary meaning of the term "data network," which is a medium that interconnects devices in a manner that enables them to send digital data packets to and receive digital data packets from each other, a Wi-Fi network is a "data network" as that term is used in claim 1. *See, e.g.,* Pl.'s Opening Markman Br. (D.I. 60 of 20-cv-881-ADA) at pp. 23-26, Exs. 24-25; Pl.'s Reply Markman Br. (D.I. 66 of 20-cv-881-ADA) at pp. 10-12, Exs. 26-27. Cast-enabled computing devices installed with various of Google's own Cast-enabled apps are programmed to perform this functionality, including but not limited to the YouTube, YouTube Music, YouTube TV, and YouTube Kids apps, as illustrated by the following screenshots:

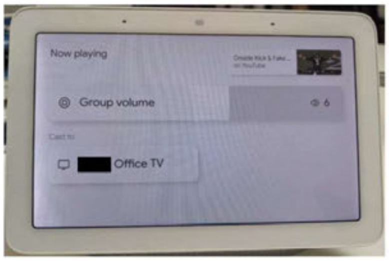
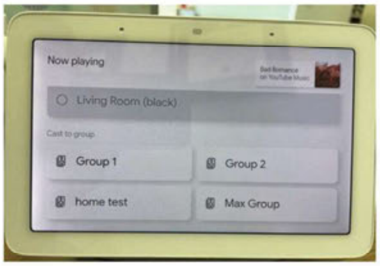
<table>
<tr><th>Claim 1</th><th>Accused Instrumentalities</th></tr>
<tr><td></td><td>

Cast to
[redacted] Office TV
Conf Room display
Link with TV code
"Baker will dethrone Big Ben" - Rich Eisen explains Why Browns will win AFC North this season
CLEVELAND BROWNS
DAILY


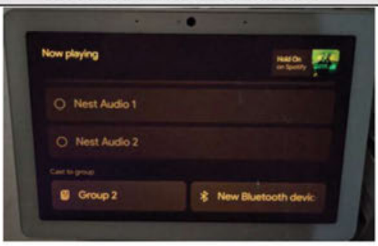




</td></tr>
</table>

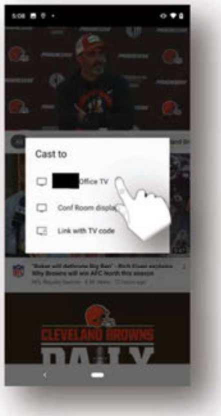
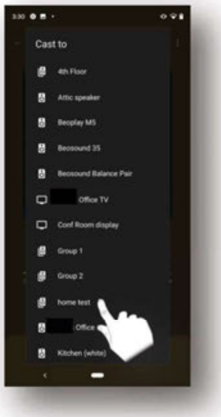
Claim 1	Accused Instrumentalities
	See also, e.g., GOOG-SONOSWDTX-00041499 [YT Orbit (MDx) Android – MediaRouter 09.30.2020]; GOOG-SONOSWDTX-00037739 [AudioPlayer + Cast + Auto] at 66-81. Likewise, Cast-enabled computing devices installed with various third-party Cast-enabled apps are also programmed to perform this functionality, including but not limited to the Spotify app, as illustrated by the following screenshot:  10:31 Listening on This phone Select a device Group 2 Google Cast Guest Room + 1 Gym Home Mini Google Cast Hub Max Google Cast Kitchen Living Room Nest Audio 1 Google Cast Nest Audio 2 Google Cast Various other Cast-enabled apps available for installation on Cast-enabled computing devices provide similar functionality. *See, e.g.,* https://support.google.com/chromecast/answer/6279384?hl=en [Cast audio from Chromecast-enabled apps to speakers]; https://www.google.com/chromecast/built-in/apps/. Each Cast-enabled display is also programmed such that, while operating in a mode in which the Cast-enabled display is configured for playback of a remote playback queue provided by a cloud-based computing system associated with a cloud-based media service (e.g., a Google service such as YouTube, YouTube Music, etc., or a third-party service such as Spotify, etc.), the Cast-enabled display is operable to detect a selection of a displayed selectable option (e.g., a selectable "Cast button") for transferring playback of audio content (e.g., music, podcasts, etc.) and/or audiovisual content (e.g., videos) from the Cast-enabled display to

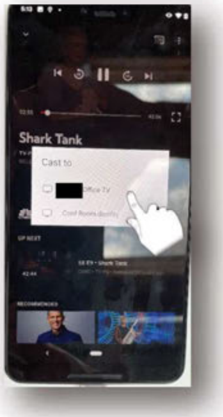
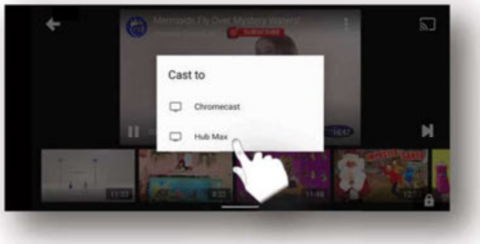
Claim 1	Accused Instrumentalities
	another device, which triggers the Cast-enabled display to display a list of available devices for transferring playback that includes one or more other Cast-enabled media players in a Cast-enabled playback system that are each (i) communicatively coupled to the Cast-enabled display over a Wi-Fi network and (ii) available to accept playback responsibility for the remote playback queue. *See, e.g.,* https://support.google.com/googlenest/answer/9563059?hl=en-IN [Move media from one cast device to another] ("At the bottom-left corner of the screen, tap Devices ⎚ to see the list of available devices and speaker groups. . . . Select the device for which you want to move your media."). Cast-enabled displays installed with various of Google's own Cast-enabled apps are programmed to perform this functionality, including but not limited to the YouTube and YouTube Music apps, as illustrated by the following photos:  Now playing Group volume 6 Cast to [redacted] Office TV  Now playing Bad Romance on YouTube Music Living Room (black) Cast to group Group 1 Group 2 home test Max Group Additionally, Cast-enabled displays installed with various third-party Cast-enabled apps are also programmed to perform this functionality, including but not limited to the Spotify app, as illustrated by the following photo:

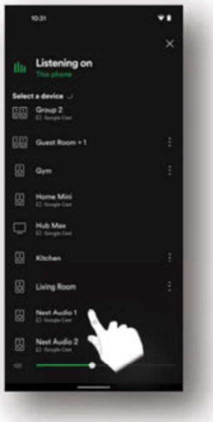
Claim 1	Accused Instrumentalities
	 Now playing Hold On on Spotify Nest Audio 1 Nest Audio 2 Cast to group Group 2 New Bluetooth devic Various other Cast-enabled apps available for installation on Cast-enabled displays provide similar functionality. As noted above, for the first time on February 3, 2022, Google proposed a construction of the term "playback queue" that would require "[a]n ordered list of multimedia items that is selected by the user for playback," and it appears Google intends to argue that the accused "Watch Next" queue is not a "remote playback queue" because it is not "[a]n ordered list of multimedia items that is selected by the user for playback." However, even if the Court were to adopt such a construction for "playback queue," Sonos maintains that the accused "Watch Next" queue would still amount to the claimed "remote playback queue" either literally or at the very least under the DoE for all of the reasons explained above in connection with claim limitation 1.4, and because of this, each Cast-enabled computing device installed with any one of the YouTube, YouTube Music, YouTube TV, or YouTube Kids apps would still satisfy claim limitation 1.5 either literally or at the very least under DoE. For instance, because a "Watch Next" queue literally amounts to a "remote playback queue" for the reasons explained above, the foregoing functionally literally amounts to the claimed functionality of claim limitation 1.5, which involves "displaying a representation of one or more playback devices in a media playback system that are each i) communicatively coupled to the computing device over a data network and ii) available to accept playback responsibility for ***the remote playback queue***." Moreover, to the extent that the Court adopts Google's construction of "playback queue" and Google then later tries to argue that a the "Watch Next" queue does not literally amount to an "ordered list of multimedia items that is selected by the user for playback" unless every single media item in the "Watch Next" queue is

Claim 1	Accused Instrumentalities
	directly selected by the user, the relevant functionality carried out by a Cast-enabled computing device with respect to a "Watch Next" queue containing at least one media item that was directly selected by a user and other items that were not directly selected by a user (e.g., media items identified by a YouTube service based on the user's selection) still satisfies claim limitation 1.5 (as well as the other claim limitations that refer to a "remote playback queue") under DoE. This is because there is merely an insubstantial difference between (i) a representation that a Cast-enabled media player is able to accept responsibility for playback of a "remote playback queue" in which all of the media items were directly selected by a user and (ii) a representation that a Cast-enabled media player is able to accept responsibility for playback of a "remote playback queue" in which only the initial media item was directly selected by the user while the other media item was identified based on the user's selection of the initial media item. Indeed, a Cast-enabled computing device performs the same function (e.g., displaying a representation of one or more Cast-enabled media players that are available to accept playback responsibility for the "Watch Next" queue), in the same way (e.g., by rendering such the representation on a display screen), to achieve the same result (e.g., presenting a user with an indication of one or more Cast-enabled media players that are available to accept playback responsibility for the "Watch Next" queue) ***regardless*** of whether the media items in the "Watch Next" queue were all directly selected by a user or only the initial media item in the "Watch Next" queue was directly selected by the user and the rest were identified based on the user's selection. Sonos further incorporates by reference Google's response to Sonos's Fact Discovery Interrogatory No. 14, including any of Google's documents or source code cited therein. *See* Google LLC's First Objections and Responses to Plaintiff Sonos, Inc.'s First Set of Fact Discovery Interrogatories.
[1.6] while displaying the representation of the one or more playback devices, receiving user input indicating a selection of at least one given playback device from the one	Each Cast-enabled control device comprises program instructions stored on the Cast-enabled control device's non-transitory computer-readable medium that, when executed by the Cast-enabled control device's processor, cause the Cast-enabled control device to, while displaying the representation of the one or more Cast-enabled media players, receive user input indicating a selection of at least one given Cast-enabled media player from the one or more Cast-enabled media players. For instance, each Cast-enabled computing device is programmed such that, while displaying the representation of the one or more Cast-enabled media players in a Cast-enabled playback system that are each on the same Wi-Fi network as the Cast-enabled computing device and available to accept playback responsibility for the remote playback queue, the Cast-enabled computing device is configured to receive user input indicating a selection of at least one Cast-enabled media player in the Cast-enabled playback system,

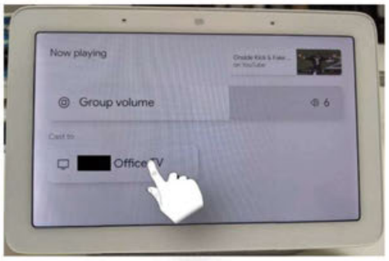
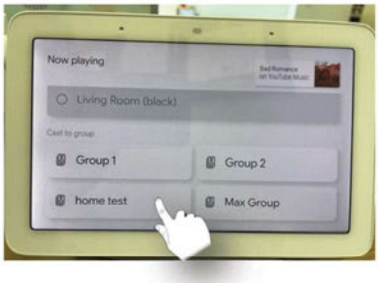
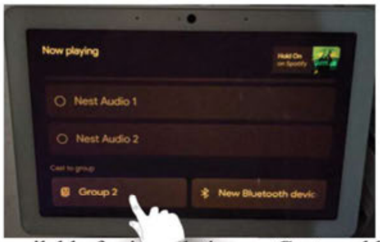
Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
or more playback devices;	which is the claimed "at least one given playback device" selected from "the one or more playback devices." *See, e.g.*, https://support.google.com/googlenest/answer/7181830 [Play media from Chromecast-enabled apps to your speaker or display] ("Tap the speaker or display for which you'd like to cast."); https://support.google.com/chromecast/answer/6279384?hl=en [Cast audio from Chromecast-enabled apps to speakers] ("3. Choose your speaker."); https://support.google.com/chromecast/answer/2995235?hl=en-AU [Cast from the YouTube app and YouTube.com] ("Tap the Chromecast device to which you want to cast."); https://support.google.com/youtubetv/answer/7353493?co=GENIE.Platform%3DAndroid&hl=en [Cast YouTube TV using Chromecast] ("4. Choose the device you want to cast to."); https://support.google.com/youtubekids/answer/6289408?hl=en&co=GENIE.Platform%3DAndroid [Watch YouTube Kids videos on your TV]; https://support.google.com/googlenest/answer/9563059?hl=en-IN [Move media from one cast device to another]. Cast-enabled computing devices installed with various of Google's own Cast-enabled apps are programmed to perform this functionality, including but not limited to the YouTube, YouTube Music, YouTube TV, and YouTube Kids apps, as illustrated by the following screenshots:  Cast to [redacted] Office TV Conf Room displa Link with TV code CLEVELAND BROWNS DAILY  Cast to 4th Floor Attic speaker Beoplay M5 Beosound 35 Beosound Balance Pair [redacted] Office TV Conf Room display Group 1 Group 2 home test [redacted] Office Kitchen (white)

Claim 1	Accused Instrumentalities
	  *See also, e.g.*, GOOG-SONOSWDTX-00041499 [YT Orbit (MDx) Android – MediaRouter 09.30.2020]; GOOG-SONOSWDTX-00037739 [AudioPlayer + Cast + Auto] at 66-81. Likewise, Cast-enabled computing devices installed with various third-party Cast-enabled apps are also programmed to perform this functionality, including but not limited to the Spotify app, as illustrated by the following screenshot:

Claim 1	Accused Instrumentalities
	 10:31 Listening on This phone Select a device Group 2 Google Cast Guest Room + 1 Gym Home Mini Google Cast Hub Max Google Cast Kitchen Living Room Nest Audio 1 Google Cast Nest Audio 2 Google Cast Various other Cast-enabled apps available for installation on Cast-enabled computing devices provide similar functionality. *See, e.g.,* https://support.google.com/chromecast/answer/6279384?hl=en [Cast audio from Chromecast-enabled apps to speakers]; https://www.google.com/chromecast/built-in/apps/. Each Cast-enabled display is also programmed such that, while displaying the representation of the one or more other Cast-enabled media players in a Cast-enabled playback system that are each on the same Wi-Fi network as the Cast-enabled display and available to accept playback responsibility for the remote playback queue, the Cast-enabled display is configured to receive user input indicating a selection of at least one other Cast-enabled media player in the Cast-enabled playback system, which is the claimed "at least one given playback device" selected from "the one or more playback devices." *See, e.g.,* https://support.google.com/googlenest/answer/9563059?hl=en-IN [Move media from one cast device to another] ("Select the device(s) to which you want to move your media."). Cast-enabled displays installed with various of Google's own Cast-enabled apps are programmed to perform this functionality, including but not limited to the YouTube and YouTube Music apps, as illustrated by the following photos:

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	 Now playing Group volume Cast to [redacted] Office TV  Now playing Living Room (black) Cast to group Group 1 Group 2 home test Max Group Additionally, Cast-enabled displays installed with various third-party Cast-enabled apps are also programmed to perform this functionality, including but not limited to the Spotify app, as illustrated by the following photo:  Now playing Hold On on Spotify Nest Audio 1 Nest Audio 2 Cast to group Group 2 New Bluetooth devic Various other Cast-enabled apps available for installation on Cast-enabled displays provide similar functionality. Sonos further incorporates by reference Google's response to Sonos's Fact Discovery Interrogatory No. 14, including any of Google's documents or source code cited therein. *See* Google LLC's First Objections and Responses to Plaintiff Sonos, Inc.'s First Set of Fact Discovery Interrogatories.
[1.7] based on receiving the user input, transmitting an instruction for	Each Cast-enabled control device comprises program instructions stored on the Cast-enabled control device's non-transitory computer-readable medium that, when executed by the Cast-enabled control device's processor, cause the Cast-enabled control device to, based on receiving the user input, transmit an instruction for the at least one given Cast-enabled media player to take over responsibility for playback of the remote

Claim 1	Accused Instrumentalities
the at least one given playback device to take over responsibility for playback of the remote playback queue from the computing device, wherein the instruction configures the at least one given playback device to (i) communicate with the cloud-based computing system in order to obtain data identifying a next one or more media items that are in the remote playback queue, (ii) use the obtained data to retrieve at least one media item in the remote playback queue from the cloud-based media service; and (iii) play back the	playback queue from the Cast-enabled control device, wherein the instruction configures the at least one given Cast-enabled media player to (i) communicate with the cloud-based computing system in order to obtain data identifying a next one or more media items that are in the remote playback queue, (ii) use the obtained data to retrieve at least one media item in the remote playback queue from the cloud-based media service, and (iii) play back the retrieved at least one media item. For instance, each Cast-enabled control device is programmed such that, based on receiving user input indicating a selection of at least one Cast-enabled media player in the Cast-enabled playback system that is on the same Wi-Fi network as the Cast-enabled control device and available to accept playback responsibility for the remote playback queue, the Cast-enabled control device is operable to transmit an instruction for the at least one Cast-enabled media player (which is the claimed "at least one given playback device") to take over responsibility for playback of the remote playback queue from the Cast-enabled computing device, where the instruction configures the at least one Cast-enabled media player to: • communicate with one or more cloud servers associated with a cloud-based media service (e.g., a Google service such as YouTube, YouTube Music, YouTube TV, YouTube Kids, etc., or a third-party service such as Spotify, etc.) that is remote from the Cast-enabled computing device and the at least one Cast-enabled media player and accessible over the Internet, which may be operated by Google or a third-party service provider, in order to obtain data identifying a next one or more media items that are in the remote playback queue (e.g., resource locators for such media items), • use the obtained data to retrieve at least one media item in the remote playback queue from the cloud-based media service; and • play back the retrieved at least one media item. *See, e.g.,* https://support.google.com/googlenest/answer/7181830 [Play media from Chromecast-enabled apps to your speaker or display]; https://support.google.com/chromecast/answer/6279384?hl=en [Cast audio from Chromecast-enabled apps to speakers]; https://support.google.com/chromecast/answer/2995235?hl=en-AU [Cast from the YouTube app and YouTube.com]; https://support.google.com/youtubetv/answer/7353493?co=GENIE.Platform%3DAndroid&hl=en [Cast YouTube TV using Chromecast]; https://support.google.com/youtubekids/answer/6289408?hl=en&co=GENIE.Platform%3DAndroid [Watch YouTube Kids videos on your TV]; https://support.google.com/googlenest/answer/9563059?hl=en-IN [Move

Claim 1	Accused Instrumentalities
retrieved at least one media item;	media from one cast device to another]; https://developers.google.com/cast/docs/web_receiver/queueing; https://developers.google.com/cast/docs/ios_sender/queueing ("The Web Receiver SDK maintains the queue and responds to operations on the queue as long as the queue has at least one item currently active (playing or paused)."); https://developers.google.com/cast/docs/android_sender/queueing ("The Receiver SDK maintains the queue and responds to operations on the queue as long as the queue has at least one item currently active (playing or paused)."). As noted above, Google's own documents acknowledge that each Cast-enabled media player (often referred to by Google as a "receiver"[10]) is programmed to maintain a local playback queue of one or more resource locators, each for multimedia content (e.g., a song or video) that is to be played back by the Cast-enabled media player. *See, e.g.*, GOOG-SONOSWDTX-00006873 [Queueing] at 74 ("Queueing allows partner applications to better integrate with Cast by providing the following features: Support of Google's and partner's cloud queue implementation so externally stored and created ***queue*** can be ***directly loaded into Cast devices***."); GOOG-SONOSWDTX-00006865 [Queueing] at 66 ("The ***Receiver*** SDK ***maintains the queue*** and responds to operations on the queue as long as the ***queue*** has ***at least one item*** currently active (playing or paused)."); https://developers.google.com/cast/docs/ios_sender/queueing ("The Web ***Receiver*** SDK ***maintains the queue*** and responds to operations on the queue as long as the ***queue*** has ***at least one item*** currently active (playing or paused)."); *see also, e.g.*, GOOG-SONOSWDTX-00006613 [Add Core Features to Your Web Receiver] at 16 ("The following media commands are currently supported in the Web Receiver SDK for Assistant-enabled devices. . . . Previous[.] Skip to the previous media item in your ***media queue***. Next[.] Skip to the next media item in your ***media queue***."), at 18 ("The Web Receiver supports preloading of

[10] *See, e.g.*, GOOG-SONOSWDTX-00006780 [Glossary] at 83 ("receiver[:] A receiver is an application created using HTML, JavaScript, and CSS. It is loaded onto a Cast device (for example, a Chromecast) through a URL that is accessible over the Wi-Fi network to which the Cast device is connected…. receiver app[:] The receiver app receives commands from the sender app and displays the requested content on the device serving as the receiver. For example, the YouTube app on Chromecast…. receiver device[:] The Cast device that loads the receiver app. For example, a Chromecast…. Web Receiver[:] A Web Receiver application is an HTML5/JavaScript application that runs on the receiver device, such as a Chromecast. It provides an interface to display the app's content on the TV, and handles messages from the sender application to control content on the receiver device."); GOOG-SONOSWDTX-00023480 [Google Cast SDK Additional Developer Terms of Service] at 81 ("A Google Web Receiver is any piece of hardware that contains Google Cast technology, including but not limited to the Chromecast device."); GOOG-SONOSWDTX-00040296 [Glossary of Terms] at 97 ("Receiver[:] A receiver device (Smart TV, Chromecast, etc.). Analogous with screen.").

Claim 1	Accused Instrumentalities
	media items after the current playback item in ***the queue***. The preload operation pre-downloads several segments of the upcoming items. The specification is done on the preloadTime value in the QueueItem object (default to 20 seconds if not provided)."); GOOG-SONOSWDTX-00006759 [GCKMediaQueueItem Class] at 63: A class representing a ***media queue item***. . . . This class is used in two-way communication between a sender application and a receiver application. The sender constructs them to ***load or insert*** a list of ***media items on the receiver application***. The GCKMediaStatus from the receiver also contains the list of items represented as instances of this class. Once loaded, the receiver will assign a unique item ID to each GCKMediaQueueItem, even if the same media gets loaded multiple times. GOOG-SONOSWDTX-00006766 [GCKMediaStatus Class] at 70 ("queueItemAtIndex: Returns the item at the specified index in the ***playback queue***. . . . queueItemWithItemID: Returns the item with the given item ID in the ***playback queue***."); GOOG-SONOSWDTX-00006780 [Glossary] at 82 ("Queueing[.] Cast utilizes both a basic sender-initiated queue and receiver-implemented queueing."); GOOG-SONOSWDTX-00006843 [Migrate to Web Receiver] at 44: This guide explains how to migrate a Cast Receiver v2 app to the latest Web Receiver app. . . . The Web Receiver API tries to follow the conventions that were introduced by CAF senders for Android and iOS, and is quite different from v2. . . . MediaManager class is replaced by PlayerManager which is a property of the CastReceiverContext singleton, and it manages the media session . . . . The PlayerManager also exposes the new sub-manager classes: . . . QueueManager - ***manage the queue***. GOOG-SONOSWDTX-00006865 [Queueing] at 66: The Cast framework provides ***queueing*** classes that support the creation of ***lists of MediaQueueItem instances***, which can be built from MediaInfo instances such as video or audio streams, ***to play sequentially on the receiver***. This ***queue of content items*** can be edited, reordered, updated, and so forth. . . . Once the last item in the queue finishes, the media session ends and the queue vanishes. GOOG-SONOSWDTX-00006878 [Queueing] at 79: The Cast framework provides ***queueing*** APIs that support the creation of ***lists of content items***, such as video or audio streams, ***to play sequentially on the Cast receiver***. The ***queue of content items*** may be edited, reordered, updated, and so forth. . . . The ***receiver*** SDK ***maintains the queue*** and responds to operations on the queue as long as the ***queue*** has ***at least one item***

Claim 1	Accused Instrumentalities
	currently active (playing or paused). . . . Once the last item in the queue finishes, the media session ends and the queue vanishes. . . . In iOS, a media queue item is represented in the Cast framework as a GCKMediaQueueItem instance. . . . Load an array of media queue items in the queue by using the appropriate queueLoadItems method of the GCKRemoteMediaClient class. GOOG-SONOSWDTX-00006953 [Web Receiver Update] at 54 ("A Web Receiver application is an HTML5/JavaScript application that runs on the ***Web Receiver device, such as a Chromecast***."), at 55: ***Queueing*** is a ***major feature*** introduced as part of ***Web Receiver***. The earlier Receiver v2 implementation carries a basic sender-initiated queue while the ***new queueing*** implementation in Web Receiver introduces ***receiver-implemented queueing***. Queueing provides the following features: Support of Google's cloud queue implementation so an externally stored and created ***queue*** can be ***directly loaded into Cast devices***. Mechanisms that allows pagination of ***items in the queue*** rather than loading everything at once, solving our v2 message size limit issue. Support for new messaging such as going to the next item, the previous item, ***fetching a window of items***, as well as getting media information related to a set of ***queue items***. Better integration with the Cast eco-system such as Google Home through ***new queueing data***. An easy to use ***QueueManager API*** that allows insertion, removal, and update of ***queue items***. GOOG-SONOSWDTX-00006965 [<GCKRemoteMediaClientListener> Protocol] at 69 ("Called when the ***media playback queue*** has been updated ***on the receiver***."); GOOG-SONOSWDTX-00007323 [Introducing Chromecast Audio] ("Chromecast Audio works with devices you already own, including Android mobile devices and tablets, iPhones and iPads . . . . [C]ontrol the speakers right from your phone – search, play, pause, ***queue songs*** and turn up the volume – anywhere in the house."); GOOG-SONOSWDTX-00025090 [com.google.android.gms.cast] at 115 ("Contains classes for interfacing with Google Cast devices. . . . The container type of the ***media queue***. . . . The type of the ***media queue***."); GOOG-SONOSWDTX-00038625 [Cast V2: Queueing Support]; GOOG-SONOSWDTX-00039091 [Cast SDK JS Team Eng Guide] at 93; GOOG-SONOSWDTX-00039480 [YouTube Music Playback Squad – Cast] at 8; GOOG-SONOSWDTX-00043603 [GPM - Chirp Integration] at 608 ("Chirp [Google Home] acts as both a Sender and Receiver . . . Queue is generated"), at 611 ("Sender asks Receiver to load the Cloud Queue and Play"); GOOG-SONOSWDTX-00051938 [YouTube Premium Subscribers Were Served in Casting] at 38-39 ("In the casting experience [sic], we are using the identity of the video adder to play the video…if a video is added by the 'Queue Autoplay' automatically, the same identity as the last video in the queue is used to play the video.").

Claim 1	Accused Instrumentalities
	Further details regarding how a Cast-enabled control device carries out this functionality, along with representative examples of evidence demonstrating this functionality, are summarized below. ***<u>YouTube, YouTube Music, YouTube TV, & YouTube Kids apps</u>*** Each Cast-enabled computing device installed with any one of the YouTube, YouTube Music, YouTube TV[11], or YouTube Kids app is programmed such that, after receiving user input indicating a selection of at least one particular Cast-enabled media player in the Cast-enabled playback system that is to take over playback responsibility, the Cast-enabled computing device functions to: • (i) instruct the particular Cast-enabled media player to launch the YouTube, YouTube Music, YouTube TV, or YouTube Kids app, which in turn causes the particular Cast-enabled media player to: • (a) contact an MDx Pairing server for a "screenID" and "lounge token"; and • (b) contact an MDx Session server with the "lounge token" to connect to an "MDx session"; • (ii) receive from the particular Cast-enabled media player an indication that the YouTube, YouTube Music, YouTube TV, or YouTube Kids app was successfully launched; • (iii) connect to the "MDx session" that the particular Cast-enabled media player connected to, which involves the Cast-enabled computing device functioning to: • (a) send to the particular Cast-enabled media player a "getMdxSessionStatus" request; • (b) receive from the particular Cast-enabled media player an "MdxSessionStatus" response containing the "screenID" that the particular Cast-enabled media player received from the MDx Pairing server; • (c) contact the MDx Pairing server for the "lounge token" that corresponds to the "screenID"; and • (d) contact the MDx Session server with the "lounge token" to connect to the "MDx session"; • (iv) transition its operating state (e.g., "playback modality") from a local playback mode to a "remote" (or "MDx") playback mode in which the Cast-enabled computing device is configured to control the

[11] For YouTube TV, the below functionality is applicable in situations where the user selects to playback multimedia content from the user's "Library" or YouTube TV's "On Demand" catalog, as opposed to from YouTube TV's "Live" content. *See, e.g.*, https://support.google.com/youtubetv/answer/7129564 [Record shows, sports, events, & movies].

Claim 1	Accused Instrumentalities
	particular Cast-enabled media player's playback of media content rather than engaging in playback of the media content itself; and • (v) transmit to one or more MDx servers (e.g., the MDx Session Server and/or a "Watch Next" server) a "setPlaylist" message that instructs the one or more MDx servers to send to the particular Cast-enabled media player a "setPlaylist" message including at least a "videoID" for a first media item and a "watchNextToken" or "WatchNextParms" for obtaining one or more next media items to be played, which in turn causes the particular Cast-enabled media player to: • (a) add a first one or more media-item identifiers (in the form of a "videoID") from a remote playback queue (e.g., a "Watch Next" queue) provided by one or more cloud servers (e.g., an "MDx," "Watch Next," or "InnerTube" server) to its local playback queue and playback a first media item: ▪ the media-item identifier of the current media item is stored in data variables including remote.ts→currentVideoIdDeprecated and remote.ts→currentWatchEndPoint.videoID[12]; ▪ the media-item identifier of the current media item is also stored in a WatchNextResponse data structure; • (b) contact the one or more cloud servers (e.g., using the "watchNextToken" or "WatchNextParms") in order to obtain data identifying a second one or more media items (each in the form of a "videoID") that are in the remote playback queue (e.g., via a "WatchNextRequest"); • (c) obtain data (in the form of a "WatchNextResponse") identifying the second one or more media items (in the form of a "videoID") that are in the remote playback queue: ▪ the "WatchNextResponse"[13] data structure on the Cast-enabled media player is defined as containing a currentVideoEndpoint data structure;[14] • Sonos notes that Google did not produce for review the source code defining currentVideoEndpoint on the Cast-enabled media player, but the code Google did produce indicates that the currentVideoEndpoint data structure contains a

[12] *See* file /2021-02-01_YTReceivers09292020/google3/video/youtube/tv/bedrock/ts/mdx/remote.ts at lines 1370-1376 (defining function getCurrentVideoId()).

[13] *See* file /2021-02-01_YTReceivers09292020/google3/video/youtube/tv/bedrock/ts/innertube/models/watch_next_response.ts.

[14] *See* file watch_next_response.ts at line 22.

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	WatchEndpoint data structure – code that Google also did not produce for review – which contains the videoID of the currently playing media item;[15] ▪ the "WatchNextResponse" data structure on the Cast-enabled media player is also defined as containing other data structures, including MdxAutoplayVideoRenderer and AutoplaySet; • Sonos notes that Google did not produce for review the source code defining MdxAutoplayVideoRenderer on the Cast-enabled media player, but the code Google did produce indicates that MdxAutoplayVideoRenderer contains a videoID for the next media-item to be played via the MDx service;[16] • Sonos also notes that Google did not produce for review the source code defining AutoplaySet on the Cast-enabled media player, but the code Google did produce indicates that AutoplaySet contains videoID values for the previous media item and next media item to be played, if applicable;[17]

[15] *See* file /2021-02-01_YTReceivers09292020/google3/video/youtube/tv/bedrock/ts/mdx/remote.ts at lines 1370-1376 (defining function getCurrentVideoId()). This function references a WatchEndpoint object called currentWatchEndpoint, which contains a videoID. The comment for this function notes that "currentWatchEndPoint should be the source of truth since that represents the current watch page." Furthermore, Google produced source code representing the "WatchEndpoint" object as used in the WatchNext service in file /2021-02-02_TYServerInnerTubeWatchNext09292020/google3/video/youtube/api/innertube/proto/navigation_end_points/watch_end_point.proto at lines 27-154 (with video_id defined at line 34).

[16] *See* file /2021-02-01_YTReceivers09292020/google3/video/youtube/tv/bedrock/ts/mdx/remote.ts at lines 2605-2610 (defining function getMdxAutoPlayVideoId()). This function references "WatchNextResponse.mdxAutoPlayVideoRenderer.videoId" at lines 2608 and 2609. Furthermore, Google produced source code representing the "MdxAutoplayVideoRenderer" object as used in the WatchNext service in file /2021-02-02_TYServerInnerTubeWatchNext09292020/google3/video/youtube/api/innertube/proto/watch_next/renderers/mdx_autoplay_video_renderer.proto at lines 24-68 (with video_id defined at line 31).

[17] *See* file /2021-02-02_TYServerInnerTubeWatchNext09292020/google3/video/youtube/api/innertube/proto/watch_next/renderers/autoplay_renderer.proto (at lines 28-124). The AutoplaySet object may have data elements for the next media-item (next_video_renderer at line 71), previous media-item (previous_video_renderer at line 75), and next auto-played media-item (autoplay_video_renderer at line 67), any of which can be of type MdxAutoplayVideoRenderer, which is known to contain a video_id.

Claim 1	Accused Instrumentalities
	▪ the media-item identifier of the next media item is stored in a data variable called upNextVideoId;[18] • (d) use the obtained data (provided in the "WatchNextResponse") to retrieve at least the second media item in the remote playback queue from the YouTube, YouTube Music, YouTube TV, or YouTube Kids media service; • (e) play back the retrieved second media item. The messages that are sent by the Cast-enabled computing device as part of this process individually or collectively amount to the claimed "instruction for the at least one given playback device to take over responsibility for playback of the remote playback queue from the computing device." Moreover, Sonos contends that the data contained in Google's "WatchNextResponse" data structure and in Google's upNextVideoId data variable, individually and collectively, amounts to the claimed "obtain[ed] data identifying a next one or more media items that are in the remote playback queue." As noted above, for the first time on February 3, 2022, Google proposed a construction of the term "playback queue" that would require "[a]n ordered list of multimedia items that is selected by the user for playback," and it appears Google intends to argue that the accused "Watch Next" queue is not a "remote playback queue" because it is not "[a]n ordered list of multimedia items that is selected by the user for playback." However, even if the Court were to adopt such a construction for "playback queue," Sonos maintains that the accused "Watch Next" queue would still amount to the claimed "remote playback queue" either literally or at the very least under the DoE for all of the reasons explained above in connection with claim limitation 1.4, and because of this, each Cast-enabled computing device installed with any one of the YouTube, YouTube Music, YouTube TV, or YouTube Kids apps would still satisfy claim limitation 1.7 either literally or at the very least under DoE. For instance, because a "Watch Next" queue literally amounts to a "remote playback queue" for the reasons explained above, the foregoing functionally literally amounts to the claimed functionality of claim limitation 1.7, which involves "transmitting an instruction for the at least one given playback device to take over

[18] *See* file /2021-02-01_YTReceivers09292020/google3/video/youtube/tv/bedrock/ts/mdx/remote.ts at lines 2605-2610 (defining function getMdxAutoPlayVideoId()).

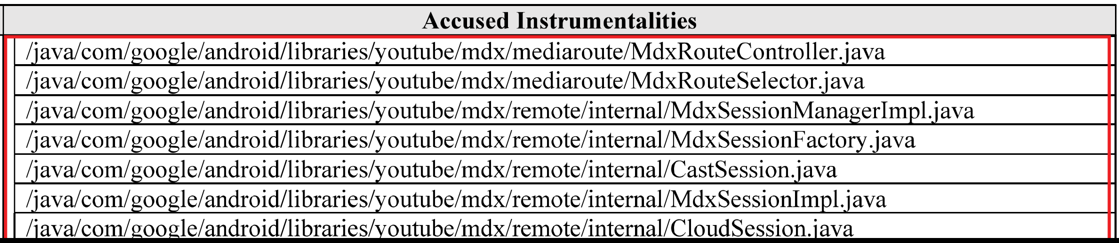
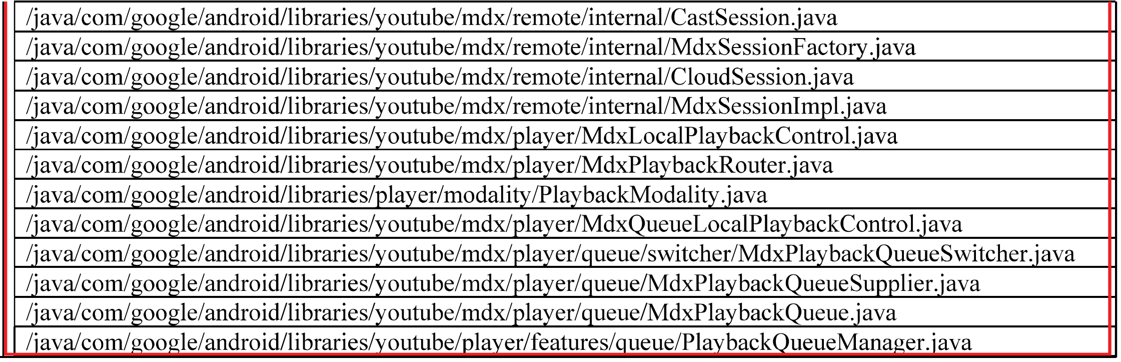
Claim 1	Accused Instrumentalities
	responsibility for playback of the ***remote playback queue*** from the computing device, wherein the instruction configures the at least one given playback device to (i) communicate with the cloud-based computing system in order to obtain data identifying a next one or more media items that are in the ***remote playback queue***, (ii) use the obtained data to retrieve at least one media item in the ***remote playback queue*** from the cloud-based media service; and (iii) play back the retrieved at least one media item." Moreover, to the extent that the Court adopts Google's construction of "playback queue" and Google then later tries to argue that a the "Watch Next" queue does not literally amount to an "ordered list of multimedia items that is selected by the user for playback" unless every single media item in the "Watch Next" queue is directly selected by the user, the relevant functionality carried out by a Cast-enabled computing device and a Cast-enabled media player with respect to a "Watch Next" queue containing at least one media item that was directly selected by a user and other items that were not directly selected by a user (e.g., media items identified by a YouTube service based on the user's selection) still satisfies claim limitation 1.7 (as well as the other claim limitations that refer to a "remote playback queue") under DoE. This is because there is merely an insubstantial difference between (i) a Cast-enabled computing device transmitting an instruction for a Cast-enabled media player to take over responsibility for playback of a "remote playback queue" in which all of the media items were directly selected by a user and (ii) a Cast-enabled computing device transmitting an instruction for a Cast-enabled media player to take over responsibility for playback of a "remote playback queue" in which only the initial media item was directly selected by the user while the other media item was identified based on the user's selection of the initial media item. Indeed, a Cast-enabled computing device performs the same function (e.g., instructing a Cast-enabled media player to take over responsibility for playback of the "Watch Next" queue), in the same way (e.g., by transmitting the instruction over a data network), to achieve the same result (e.g., configuring the Cast-enabled media player to perform the playback-device functions recited in claim limitation 1.7) ***regardless*** of whether the media items in the "Watch Next" queue were all directly selected by a user or only the initial media item in the "Watch Next" queue was directly selected by the user and the rest were identified based on the user's selection. And likewise, a Cast-enabled media player performs the same functions (e.g., obtaining data identifying a next one or more media items that are in the "Watch Next" queue and using the obtained data to retrieve at least one media item in the "Watch Next" queue from the cloud-based media service), in the same way (e.g., by communicating with one or more cloud servers), to achieve the same result (e.g., assuming responsibility for playback of the "Watch Next" queue) ***regardless*** of whether the media items in the "Watch Next" queue were

Claim 1	Accused Instrumentalities
	all directly selected by a user or only the initial media item in the "Watch Next" queue was directly selected by the user and the rest were identified based on the user's selection. The following exemplary evidence demonstrates that each Cast-enabled control device installed with the YouTube, YouTube Music, YouTube TV, or YouTube Kids app is programmed to perform this functionality: • GOOG-SONOSWDTX-00037243 [MDX Communication Protocol v3 07.03.2018] at 51-52, 58-59; • GOOG-SONOSWDTX-00039484 [YT Orbit – Cast 10.08.2020] at 85-86; • GOOG-SONOSWDTX-00039491 [YT Orbit (MDx) – Credential Transfer Tokens (CTTs) 10.08.2020] at 91: ◦ "When Kabuki makes a playback request, it includes both a video id and a CTT. This allows the video to play as if the user were logged in to Kabuki (even if another user is already logged in to Kabuki). When Kabuki plays a video in the queue, the Watch Next service will send the video id and CTT (if any) for the next video in the queue so that Kabuki knows what to play when the current video finishes." ◦ "The session server then sends the first video and its associated CTT to Kabuki. The session server also sends the shared queue id to Kabuki . . . . Kabuki then requests a bunch of services using the videoId and the token. When making a request to the Watch Next service, Kabuki also passes along the shared queue id so that the Watch Next service can ask the session server to return the next (videoId, CTT) pair in the shared queue." • GOOG-SONOSWDTX-00039494 [YT Orbit – DIAL 10.08.2020] at 94-96; • GOOG-SONOSWDTX-00039511 [YT Orbit (MDx) – InnerTube Interactions 10.08.2020] at 11; • GOOG-SONOSWDTX-00039785 [YTM Playback Squad – Server 01.05.2021] at 85, 89-90; • GOOG-SONOSWDTX-00039813 [YT-Orbit – Cast] at 13; • GOOG-SONOSWDTX-00039819 [YT-Orbit – Session Server]; • GOOG-SONOSWDTX-00039916 [MDx Overview] at 89 ("When Kabuki plays a video in the queue, the Watch Next service will send the video id and CTT (if any) for the next video in the queue so that Kabuki knows what to play when the current video finishes."), at 91; • GOOG-SONOSWDTX-00040156 [YT Orbit Servers Documentation – YouTube Orbit Servers 03.25.2020]; • GOOG-SONOSWDTX-00040283 [YT-Orbit – CTT Service] at 84;

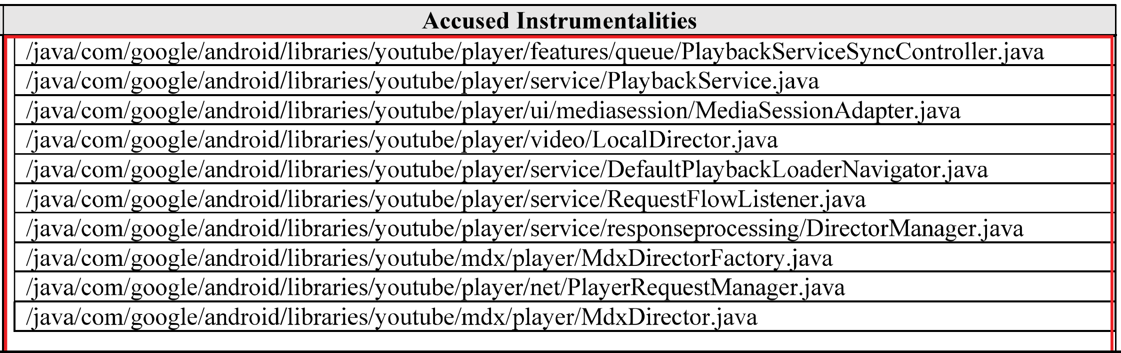
Claim 1	Accused Instrumentalities
	• GOOG-SONOSWDTX-00040287 [YT-Orbit – Credential Transfer Tokens (CTTS)] at 87; • GOOG-SONOSWDTX-00040290 [YT-Orbit – DIAL (Discovery and Launch)] at 91; • GOOG-SONOSWDTX-00040317 [YT-Orbit – Session Service]; • GOOG-SONOSWDTX-00040622 [WatchNext – Introduction 07.15.2020]; • GOOG-SONOSWDTX-00041617 [YouTube Music Playback History in MDx Proposal] • "MDx Session Server informs the TV to initiate video playback for the first video in the album, and points it to the shared queue to retrieve the remaining videoIds to play next." • "MDx Session server sends the videoEntry to Kabuki to inform the TV to initiate video playback for the first video in the album, and points it to the shared queue to retrieve the remaining videoEntrys to play next." • "When the next video in the MDx queue is to be played, Kabuki calls the WatchNext service to retrieve the next video from the shared queue." • GOOG-SONOSWDTX-00041241 [WatchNext Suggestions – Watch Next Suggestions 07.16.2020]; • GOOG-SONOSWDTX-00041491 [YT Orbit (MDx) Android – MDx Playback on Android 01.12.2021] at 93-95; • GOOG-SONOSWDTX-00041499 [YT Orbit (MDx) Android – MediaRouter 09.30.2020] at 499-500; • GOOG-SONOSWDTX-00041934 [Cast Stream Transfer and YouTube 06.04.2018] at 35; • GOOG-SONOSWDTX-00041968 [Credential Transfer Tokens(go/mdxctt) 11.18.2014]; • GOOG-SONOSWDTX-00050998 [Home Group Determination for YTM Audio Tier 10.24.2018] at 1000-1001; • GOOG-SONOSWDTX-00051490 [YTM Cast: Loop, aka Repeat 11.03.2020] at 90; • GOOG-SONOSWDTX-00052083 [YouTube Developer's Handbook – Life of a YouTube upload 01.28.2021] at 96-97; • GOOG-SONOSWDTX-00052121 [YouTube Developer's Handbook – Life of a video watch 01.11.2021] at 123-126. Representative excerpts of Google's YouTube app source code[19] related to the aforementioned functionality include:

[19] Root directory: /2020-09-22-youtube_android_15.38.35/

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	 /java/com/google/android/libraries/youtube/mdx/mediaroute/MdxRouteController.java /java/com/google/android/libraries/youtube/mdx/mediaroute/MdxRouteSelector.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionManagerImpl.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionFactory.java /java/com/google/android/libraries/youtube/mdx/remote/internal/CastSession.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionImpl.java /java/com/google/android/libraries/youtube/mdx/remote/internal/CloudSession.java [redacted]  /java/com/google/android/libraries/youtube/mdx/remote/internal/CastSession.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionFactory.java /java/com/google/android/libraries/youtube/mdx/remote/internal/CloudSession.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionImpl.java /java/com/google/android/libraries/youtube/mdx/player/MdxLocalPlaybackControl.java /java/com/google/android/libraries/youtube/mdx/player/MdxPlaybackRouter.java /java/com/google/android/libraries/player/modality/PlaybackModality.java /java/com/google/android/libraries/youtube/mdx/player/MdxQueueLocalPlaybackControl.java /java/com/google/android/libraries/youtube/mdx/player/queue/switcher/MdxPlaybackQueueSwitcher.java /java/com/google/android/libraries/youtube/mdx/player/queue/MdxPlaybackQueueSupplier.java /java/com/google/android/libraries/youtube/mdx/player/queue/MdxPlaybackQueue.java /java/com/google/android/libraries/youtube/player/features/queue/PlaybackQueueManager.java

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	 /java/com/google/android/libraries/youtube/player/features/queue/PlaybackServiceSyncController.java /java/com/google/android/libraries/youtube/player/service/PlaybackService.java /java/com/google/android/libraries/youtube/player/ui/mediasession/MediaSessionAdapter.java /java/com/google/android/libraries/youtube/player/video/LocalDirector.java /java/com/google/android/libraries/youtube/player/service/DefaultPlaybackLoaderNavigator.java /java/com/google/android/libraries/youtube/player/service/RequestFlowListener.java /java/com/google/android/libraries/youtube/player/service/responseprocessing/DirectorManager.java /java/com/google/android/libraries/youtube/mdx/player/MdxDirectorFactory.java /java/com/google/android/libraries/youtube/player/net/PlayerRequestManager.java /java/com/google/android/libraries/youtube/mdx/player/MdxDirector.java [redacted]

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	[redacted] Representative excerpts of Google's YouTube Music app source code[20] related to the aforementioned functionality include: /java/com/google/android/libraries/youtube/mdx/mediaroute/MdxRouteController.java /java/com/google/android/libraries/youtube/mdx/mediaroute/MdxRouteSelector.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionManagerImpl.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionFactory.java

[20] Root directory: /2020-10-08-youtube-music_3.87.53/

Claim 1	Accused Instrumentalities
	/java/com/google/android/libraries/youtube/mdx/remote/internal/CastSession.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionImpl.java /java/com/google/android/libraries/youtube/mdx/remote/internal/CloudSession.java [redacted] /java/com/google/android/libraries/youtube/mdx/remote/internal/CastSession.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionFactory.java /java/com/google/android/libraries/youtube/mdx/remote/internal/CloudSession.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionImpl.java /java/com/google/android/libraries/youtube/mdx/player/MdxLocalPlaybackControl.java /java/com/google/android/libraries/youtube/mdx/player/MdxPlaybackRouter.java /java/com/google/android/libraries/player/modality/PlaybackModality.java /java/com/google/android/libraries/youtube/mdx/player/MdxQueueLocalPlaybackControl.java /java/com/google/android/libraries/youtube/mdx/player/queue/switcher/MdxPlaybackQueueSwitcher.java /java/com/google/android/libraries/youtube/mdx/player/queue/MdxPlaybackQueueSupplier.java /java/com/google/android/libraries/youtube/mdx/player/queue/MdxPlaybackQueue.java /java/com/google/android/libraries/youtube/player/features/queue/PlaybackQueueManager.java /java/com/google/android/libraries/youtube/player/features/queue/PlaybackServiceSyncController.java /java/com/google/android/libraries/youtube/player/service/PlaybackService.java /java/com/google/android/libraries/youtube/player/ui/mediasession/MediaSessionAdapter.java /java/com/google/android/libraries/youtube/player/video/LocalDirector.java

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	/java/com/google/android/libraries/youtube/player/service/DefaultPlaybackLoaderNavigator.java /java/com/google/android/libraries/youtube/player/service/RequestFlowListener.java /java/com/google/android/libraries/youtube/player/service/responseprocessing/DirectorManager.java /java/com/google/android/libraries/youtube/mdx/player/MdxDirectorFactory.java /java/com/google/android/libraries/youtube/player/net/PlayerRequestManager.java /java/com/google/android/libraries/youtube/mdx/player/MdxDirector.java [redacted]

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	[redacted] Representative excerpts of Google's YouTube TV app source code[21] related to the aforementioned functionality include: /java/com/google/android/libraries/youtube/mdx/mediaroute/MdxRouteController.java /java/com/google/android/libraries/youtube/mdx/mediaroute/MdxRouteSelector.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionManagerImpl.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionFactory.java /java/com/google/android/libraries/youtube/mdx/remote/internal/CastSession.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionImpl.java /java/com/google/android/libraries/youtube/mdx/remote/internal/CloudSession.java [redacted]

[21] Root directory: /2020-09-29-youtube_tv_4.38.3/

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	[redacted] /java/com/google/android/libraries/youtube/mdx/remote/internal/CastSession.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionFactory.java /java/com/google/android/libraries/youtube/mdx/remote/internal/CloudSession.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionImpl.java /java/com/google/android/libraries/youtube/mdx/player/MdxLocalPlaybackControl.java /java/com/google/android/libraries/youtube/mdx/player/MdxPlaybackRouter.java /java/com/google/android/libraries/youtube/player/modality/PlaybackModality.java /java/com/google/android/libraries/youtube/mdx/player/MdxQueueLocalPlaybackControl.java /java/com/google/android/libraries/youtube/mdx/player/queue/switcher/MdxPlaybackQueueSwitcher.java /java/com/google/android/libraries/youtube/mdx/player/queue/MdxPlaybackQueueSupplier.java /java/com/google/android/libraries/youtube/mdx/player/queue/MdxPlaybackQueue.java /java/com/google/android/libraries/youtube/player/features/queue/PlaybackQueueManager.java /java/com/google/android/libraries/youtube/player/features/queue/PlaybackServiceSyncController.java /java/com/google/android/libraries/youtube/player/service/PlaybackService.java /java/com/google/android/libraries/youtube/player/ui/mediasession/MediaSessionAdapter.java /java/com/google/android/libraries/youtube/player/video/LocalDirector.java /java/com/google/android/libraries/youtube/player/service/DefaultPlaybackLoaderNavigator.java /java/com/google/android/libraries/youtube/player/service/RequestFlowListener.java /java/com/google/android/libraries/youtube/player/service/responseprocessing/DirectorManager.java /java/com/google/android/libraries/youtube/mdx/player/MdxDirectorFactory.java /java/com/google/android/libraries/youtube/player/net/PlayerRequestManager.java /java/com/google/android/libraries/youtube/mdx/player/MdxDirector.java

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	[redacted]

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	[redacted] Representative excerpts of Google's YouTube Kids app source code[22] related to the aforementioned functionality include: /java/com/google/android/libraries/youtube/mdx/mediaroute/MdxRouteController.java /java/com/google/android/libraries/youtube/mdx/mediaroute/MdxRouteSelector.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionManagerImpl.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionFactory.java /java/com/google/android/libraries/youtube/mdx/remote/internal/CastSession.java /java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionImpl.java /java/com/google/android/libraries/youtube/mdx/remote/internal/CloudSession.java [redacted]

[22] Root directory: 2020-09-28-youtube_kids_5.43.3/

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities

	▪ MdxSessionFactory.createCloudDelegate() // 267-276
	• MdxSessionFactory.makeCloudSesssion() // 278-311
	• MdxSessionImpl.launchDelegate() // 258-269
	o MdxSessionImpl.launchApp() // 331-336
	▪ CloudSession.doLaunchApp() // 394-401
	/java/com/google/android/libraries/youtube/mdx/remote/internal/CastSession.java
	/java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionFactory.java
	/java/com/google/android/libraries/youtube/mdx/remote/internal/CloudSession.java
	/java/com/google/android/libraries/youtube/mdx/remote/internal/MdxSessionImpl.java
	/java/com/google/android/libraries/youtube/mdx/player/MdxLocalPlaybackControl.java
	/java/com/google/android/libraries/youtube/mdx/player/MdxPlaybackRouter.java
	/java/com/google/android/libraries/youtube/player/modality/PlaybackModality.java
	/java/com/google/android/libraries/youtube/mdx/player/MdxQueueLocalPlaybackControl.java
	/java/com/google/android/libraries/youtube/mdx/player/queue/switcher/MdxPlaybackQueueSwitcher.java
	/java/com/google/android/libraries/youtube/mdx/player/queue/MdxPlaybackQueueSupplier.java
	/java/com/google/android/libraries/youtube/mdx/player/queue/MdxPlaybackQueue.java
	/java/com/google/android/libraries/youtube/player/features/queue/PlaybackQueueManager.java
	/java/com/google/android/libraries/youtube/player/features/queue/PlaybackServiceSyncController.java
	/java/com/google/android/libraries/youtube/player/service/PlaybackService.java
	/java/com/google/android/libraries/youtube/player/ui/mediasession/MediaSessionAdapter.java
	/java/com/google/android/libraries/youtube/player/video/LocalDirector.java
	/java/com/google/android/libraries/youtube/player/service/DefaultPlaybackLoaderNavigator.java
	/java/com/google/android/libraries/youtube/player/service/RequestFlowListener.java
	/java/com/google/android/libraries/youtube/player/service/responseprocessing/DirectorManager.java
	/java/com/google/android/libraries/youtube/mdx/player/MdxDirectorFactory.java
	/java/com/google/android/libraries/youtube/player/net/PlayerRequestManager.java
	/java/com/google/android/libraries/youtube/mdx/player/MdxDirector.java
	[redacted]

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	[redacted]

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	[redacted]
	[redacted] Representative excerpts of Google's server source code related to the aforementioned functionality include: [23]/google3/java/com/google/youtube/lounge/browserchannel/RealLoungeSessionManager.java /google3/java/com/google/youtube/lounge/browserchannel/LoungeSessionSharedQueueExpander.java /google3/java/com/google/youtube/lounge/browserchannel/LoungeSession.java /google3/java/com/google/youtube/lounge/browserchannel/LoungeMessageSender.java /google3/java/com/google/youtube/lounge/browserchannel/RealMessageSender.java /google3/java/com/google/youtube/lounge/browserchannel/LoungeChannel.java [redacted]

[23] Root directory: /2021-02-01_YTServerMDx09292020/

Claim 1	Accused Instrumentalities
	[24]/google3/video/youtube/api/innertube/proto/innertube_service.proto /google3/video/youtube/api/innertube/proto/watch_next/services/innertube_watch_next_service.proto /google3/video/youtube/src/python/servers/innertube/watch_next/innertube_watch_next.py /google3/video/youtube/src/python/servers/innertube/watch_next/_content.py /google3/video/youtube/src/python/servers/innertube/watch_next/navigation_list.py [redacted] Representative excerpts of Google's Cast-enabled media player source code[25] related to the aforementioned functionality include: /google3/video/youtube/web/living_room/contrib/cast/mdx_session/cast_mdx_session_service.ts

[24] Root directory: /2021-02-02_YTServerInnerTubeWatchNext09292020/
[25] Root directory: /2021-02-01_YTReceivers09292020/

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	/google3/video/youtube/src/web/javascript/library/mdx/screen_ts/loungeadapter.ts /google3/video/youtube/tv/bedrock/ts/mdx/services/remote.ts /google3/video/youtube/tv/bedrock/ts/mdx/models/remote_queue_metadata.ts /google3/video/youtube/tv/bedrock/ts/watch/mdx/remote_plugin.ts [redacted] /google3/video/youtube/web/player/videodatafetch.js /google3/video/youtube/web/player/application.ts [redacted]

Claim 1	Accused Instrumentalities
	[redacted] Sonos further incorporates by reference Google's response to Sonos's Fact Discovery Interrogatory Nos. 14-15, including any of Google's documents or source code cited therein. *See* Google LLC's First Objections and Responses to Plaintiff Sonos, Inc.'s First Set of Fact Discovery Interrogatories. ***<u>Spotify app</u>*** Each Cast-enabled control device installed with the Spotify app (which is Cast-enabled and utilizes the Cast SDK) is programmed such that, after receiving user input indicating a selection of at least one particular Cast-enabled media player in the Cast-enabled playback system that is to take over playback responsibility, the Cast-enabled computing device functions to: • (i) instruct the particular Cast-enabled media player to launch the Spotify app and "connect" to the Cast-enabled computing device; • (ii) instruct the particular Cast-enabled media player to takeover playback of a remote queue (e.g., a "Spotify queue") from the Cast-enabled computing device (e.g., via a "LoadRequest"), which causes the particular Cast-enabled media player to: • (a) communicate with one or more cloud servers providing the remote queue previously being played by the Cast-enabled computing device to obtain from the one or more cloud servers one or more media-item identifiers (e.g., "contentIds") from the remote queue and playback a first media item; • (b) before finishing playback of the first media item, communicate with the one or more cloud servers to obtain data identifying a next one or more media items that are in the remote playback queue (e.g., "Spotify queue"); • (c) use the obtained data to retrieve at least one media item in the remote playback queue from the Spotify media service; and • (d) play back the retrieved at least one media item;

Claim 1	Accused Instrumentalities
	• (iii) detect an indication that the particular Cast-enabled media player has taken over playback responsibility for the remote queue; and • (iv) transition its operating state from a local playback mode to a remote playback mode (e.g., switch its "control category" to "remote playback") in which the Cast-enabled computing device is configured to control the particular Cast-enabled media player's playback of media content rather than engaging in playback of the media content itself. The messages that are sent by the Cast-enabled computing device as part of this process individually or collectively amount to the claimed "instruction for the at least one given playback device to take over responsibility for playback of the remote playback queue from the computing device." The following exemplary evidence demonstrates that each Cast-enabled control device installed with the Spotify app is programmed with this functionality: • GOOG-SONOSWDTX-00038459 [Cast Sender Android SDK v2 Architecture] at 59 ("Once a device is discovered, the application may use either the MediaRouter media control APIs, or the Cast API to interact with the device. . . . The Cast Device Controller is a service-side component that implements all of the logic for interacting with a Cast device. This includes managing the connection to the device . . . launching and stopping receiver applications on the device . . . ."), at 61 ("In addition to implementing the standard 'remote media playback' facilities of the Media Router framework, [the Cast Media Route Provider] also implements some Cast-specific extensions . . . ."), at 65-66 (describing "remote playback" "control category"); • https://developers.google.com/cast/docs/web_receiver/queueing?hl=en ("Queueing allows partner applications to better integrate with Cast by providing the following features: Support of Google's and partner's cloud queue implementation so externally stored and created queue can be directly loaded into Cast devices."); • https://developers.google.com/cast/docs/reference/web_sender/chrome.cast.media.MediaInfo; • https://developers.google.com/cast/docs/reference/web_sender/chrome.cast.media.LoadRequest. Sonos further incorporates by reference Google's response to Sonos's Fact Discovery Interrogatory Nos. 14-15, including any of Google's documents or source code cited therein. *See* Google LLC's First Objections and Responses to Plaintiff Sonos, Inc.'s First Set of Fact Discovery Interrogatories.

Claim 1	Accused Instrumentalities
	Cast-Enabled Displays Each Cast-enabled display is programmed such that, after receiving user input indicating a selection of at least one particular Cast-enabled media player in the Cast-enabled playback system that is to take over playback responsibility, the Cast-enabled display functions to: • (i) instruct the particular Cast-enabled media player to launch whichever app the Cast-enabled display was playing audio content (e.g., music, podcasts, etc.) and/or audiovisual content (e.g., videos) ~~content~~ from when it detected the set of inputs (~~e.g.,~~ via a “launch” message that gets transmitted over Wi-Fi after the Cast-enabled display processes an internal “set playback devices” message and an internal “StoreSession” message); • (ii) transfer the state of the Cast-enabled display's current playback session of a remote queue to the particular Cast-enabled media player (~~e.g.,~~ via a ~~“Load” and/or~~ “ResumeSession” message), which in turn causes the particular Cast-enabled media player to: • (a) communicate with one or more cloud servers providing the remote queue previously being played by the Cast-enabled display to obtain from the one or more cloud servers one or more media-item identifiers (e.g., “contentIds”) from the remote queue and playback a first media item; • (b) before finishing playback of the first media item, communicate with the one or more cloud servers to obtain data identifying a next one or more media items that are in the remote playback queue; • (c) use the obtained data to retrieve at least one media item in the remote playback queue from the same streaming content service that the Cast-enabled display was playing back from when it received the user input; and • (d) play back the retrieved at least one media item; • (iii) detect an indication that the particular Cast-enabled media player has taken over playback responsibility for the remote queue; and • (iv) transition its operating state from a local playback mode to a remote playback mode in which the Cast-enabled computing device is configured to control the particular Cast-enabled media player's playback of media content rather than engaging in playback of the media content itself.

Claim 1	Accused Instrumentalities
	The messages that are sent by the Cast-enabled display as part of this process individually or collectively amount to the claimed "instruction for the at least one given playback device to take over responsibility for playback of the remote playback queue from the computing device." The following exemplary evidence demonstrates that each Cast-enabled display is programmed with this functionality: • https://developers.google.com/cast/docs/web_receiver/core_features ("Stream transfer[.] Preserving session state is the basis of stream transfer, a CAF feature where users can move existing audio and video streams across devices using … smart displays. Media stops playing on one device (the source) and continues on another (the destination)…. The event flow for stream transfer is: 1. On the source device: a. Media stops playing. b. The Web Receiver application receives a command to save the current media state. c. The Web Receiver application is shut down. 2. On the destination device: a. The Web Receiver application is loaded. b. The Web Receiver application receives a command to restore the saved media state. c. Media resumes playing. Elements of media state include: • Specific position or timestamp of the song, video, or media item. • Its place in a broader queue (such as a playlist or artist radio). • The authenticated user. • Playback state (for example, playing or paused)."), • *Id.* ("Preserving session state[.] The Web Receiver SDK provides a default implementation for Web Receiver apps to preserve session states by taking a snapshot of current media status, converting the status into a load request, and resuming the session with the load request."); • https://developers.google.com/cast/docs/reference/web_receiver/cast.framework.messages.LoadRequestData [Class: LoadRequestData]; • https://developers.google.com/cast/docs/reference/web_sender/chrome.cast.media.MediaInfo [Class: MediaInfo];

Claim 1	Accused Instrumentalities
	• GOOG-SONOSWDTX-00008248 [Move your music between rooms with stream transfer 10.08.2019] at 48 ("Stream transfer is a new feature that lets you easily move music, videos, podcasts and more between compatible devices in your home using . . . the touchscreen on your Nest smart display. . . . **Move YouTube videos between your Nest smart display and Chromecast-enabled TV:** Browse for your favorite YouTube videos on Nest Hub Max, and tap the cast control on the screen to move it to your Chromecast-connected TV. . . . Transfer music from a single speaker to the speaker group to fill your whole home with music. . . . Stream transfer is compatible with your favorite audio apps, including YouTube Music, Spotify, Pandora, and more. For video, you can enjoy the millions of videos available on YouTube."); • GOOG-SONOSWDTX-00049676 [Stream Expansion/Transfer V2 Protocols 05.17.2018] at 77 ("Sender sends [SET_PLAYBACK_DEVICES] request to any groupable endpoint to configure which devices are party of current playback session, or to change the session owner (stream transfer). . . . For stream transfer, sender simply gives details of new session owner. Sender will get back a PlaybackSessionUpdated message with matching request ID on success. . . . If the playback session is moved to a new endpoint, the PlaybackSessionUpdated response (which notifies success) will contain the endpoint details. Sender should migrate to that new endpoint."), at 77-79, 83 • GOOG-SONOSWDTX-00038533 [Cast V2: Application Protocol 06.29.2017] at 54-55 ("contentId" "is the service-specific identifier of the content currently loaded by the media player. . . . In most cases, this will be the URL to the media . . . ."), at 65-66 (setting forth parameters of "Load" message), at 74-75 (setting forth parameters of "ResumeSession" message); • GOOG-SONOSWDTX-00041934 [Cast Stream Transfer and YouTube 06.04.2018] at 35 ("Video device: Chromecast (normal or ultra) or Dragonglass device . . . . Transfer a music stream from a video device to an audio device . . . . Transfer a video stream between 2 video devices"), at 38-39 (describing process of transferring "CloudSession" from first Kabuki device to second Kabuki device, which involves transferring information regarding "[c]urrent playlistId (The remote queue RQ)" and causing MDx server to "[k]eep alive the RQ playlist and the CTT associated with it in the case of a steram transfer."); • GOOG-SONOSWDTX-00044545 [Stream Transfer Technical Design 03.21.2018] at 46-54; • GOOG-SONOSWDTX-00049687 [Stream transfer for cast sender apps] at 88; • GOOG-SONOSWDTX-00048872 [Home Device Stream Transfer: PRD 03.05.2018] at 75; • GOOG-SONOSWDTX-00052083 [YouTube Developer's Handbook – Life of a YouTube upload 01.28.2021] at 96-97;

Claim 1	Accused Instrumentalities
	• GOOG-SONOSWDTX-00039511 [YT Orbit (MDx) – InnerTube Interactions 10.08.2020] at 11; • GOOG-SONOSWDTX-00039785 [YTM Playback Squad – Server 01.05.2021] at 85, 89-90; • GOOG-SONOSWDTX-00050998 [Home Group Determination for YTM Audio Tier 10.24.2018] at 1000-1001. Representative excerpts of Google's Cast-enabled display source code[26] related to the aforementioned functionality include: /assistant/display/cast/media/media_session_manager.ts [redacted] Representative excerpts of Google's Cast-enabled media player source code[27] related to the aforementioned functionality include: /chrome/dongle/receiver/js/medianamespace.js /chrome/dongle/receiver/js/mediamanager.js [redacted]

[26] Root directory: /2020-09-01-google3/

[27] Root directory: /2020-09-01-google3/

Claim 1	Accused Instrumentalities
	[redacted] Additional exemplary source code demonstrating that a Cast-enabled display is programmed with the aforementioned functional capability include: /chromecast-internal-1.50/receiver/cast_session/cast_session_manager.cc /chromecast-internal-1.50/multizone/manager/multizone_namespace_handler.cc /chromecast-internal-1.50/multizone/manager/multizone_group.cc /chromecast-internal-1.50/multizone/manager/multizone_manager.cc /chromecast-internal-1.50/multizone/manager/dynamic_group_manager.cc /chromecast-internal-1.50/multizone/manager/stream_transfer_manager.cc [redacted]

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	/chrome/dongle/receiver/js/mediamanger.js [28]/google3/video/youtube/src/web/javascript/library/mdx/screen_ts/castadapter.ts /google3/video/youtube/tv/bedrock/ts/mdx/services/remote.ts [redacted] /chromecast-internal-1.50/multizone/manager/stream_transfer_manager.cc /chromecast-internal-1.50/multizone/manager/multizone_manager.cc /chromecast-internal-1.50/multizone/manager/stream_transfer_connecton.cc /chromecast-internal-1.50/receiver/net/remote_connection_impl.cc /chromecast-internal-1.50/receiver/net/remote_control_socket.cc [redacted]

[28] Root directory: /2021-02-01_YTReceivers09292020/

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	[redacted]
	[redacted] /chromecast-internal-1.50/multizone/manager/stream_transfer_connection.cc /chromecast-internal-1.50/receiver/app/receiver_namespace_handler.cc /chromecast-internal-1.50/receiver/net/remote_connection_impl.cc [redacted] /chromecast-internal-1.50/receiver/net/remote_connection_impl.cc /chromecast-internal-1.50/multizone/manager/stream_transfer_connection.cc /chromecast-internal-1.50/receiver/app/receiver_namespace_handler.cc [redacted]

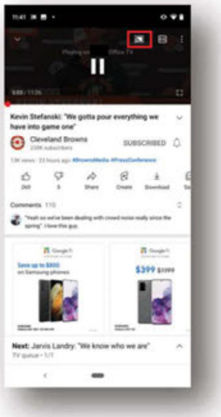
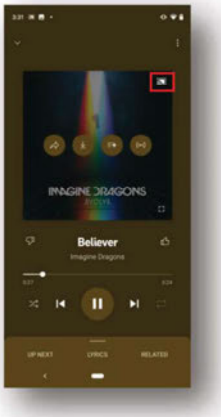
Claim 1	Accused Instrumentalities
	/chromecast-internal-1.50/multizone/manager/stream_transfer_connection.cc /chromecast-internal-1.50/receiver/net/remote_connection_impl.cc /chromecast-internal-1.50/multizone/manager/stream_transfer_manager.cc [redacted] Additional exemplary source code of Google's Cast-enabled media player source code[29] related to the aforementioned functionality include: /chromecast-internal-1.50/receiver/app/receiver_namespace_handler.cc /chromecast-internal-1.50/receiver/app/application_manager_impl.cc /chrome/dongle/receiver/js/mediamanager.js [30]/google3/video/youtube/src/web/javascript/library/mdx/screen_ts/castadapter.ts /google3/video/youtube/tv/bedrock/ts/mdx/services/remote.ts [redacted]

[29] Root directory: /2020-09-01-google3/

[30] Root directory: /2021-02-01_YTReceivers09292020/

Claim 1	**Accused Instrumentalities**
	[redacted] Sonos further incorporates by reference Google's response to Sonos's Fact Discovery Interrogatory Nos. 14-15, including any of Google's documents or source code cited therein. *See* Google LLC's First Objections and Responses to Plaintiff Sonos, Inc.'s First Set of Fact Discovery Interrogatories.
[1.8] detecting an indication that playback responsibility for the remote playback queue has been successfully transferred from the computing device to the at least one given playback device;	Each Cast-enabled control device comprises program instructions stored on the Cast-enabled control device's non-transitory computer-readable medium that, when executed by the Cast-enabled control device's processor, cause the Cast-enabled control device to detect an indication that playback responsibility for the remote playback queue has been successfully transferred from the Cast-enabled control device to the at least one given Cast-enabled media player. For instance, each Cast-enabled computing device is programmed with the capability to detect an indication that playback responsibility for the remote playback queue has been successfully transferred from the Cast-enabled computing device to at least one Cast-enabled media player (which is the claimed "at least one given playback device"), which is demonstrated by the fact that the Cast-enabled computing device displays an indicator that playback responsibility for the remote playback queue has been successfully transferred to the at least one Cast-enabled media player that takes the form of a "Cast button" that is "filled in" or turns "dark grey" (among other factors confirming that the Cast-enabled computing device has detected an indication that playback responsibility for the remote playback queue has been successfully transferred to at least one Cast-enabled media player). *See, e.g.,* https://support.google.com/googlenest/answer/7181830 [Play media from Chromecast-enabled apps to your speaker or display] ("When you're connected, the Cast button will turn from light to dark grey, letting you know that you're connected."); https://support.google.com/chromecast/answer/6279384?hl=en [Cast audio from Chromecast-enabled apps to speakers] ("The Cast button will change colors , letting you know you're connected."); https://support.google.com/chromecast/answer/2995235?hl=en-AU [Cast from the YouTube app and YouTube.com]; https://support.google.com/youtubetv/answer/7353493?co=GENIE.Platform%3DAndroid&hl=en [Cast YouTube TV using Chromecast];

Ex. B –Infringement Contention Chart: U.S. Patent No. 10,779,033
HIGHLY CONFIDENTIAL - SOURCE CODE - ATTORNEYS' EYES ONLY

Claim 1	Accused Instrumentalities
	https://support.google.com/youtubekids/answer/6289408?hl=en&co=GENIE.Platform%3DAndroid [Watch YouTube Kids videos on your TV]; https://support.google.com/chromecast/answer/3265953?hl=en [Chromecast-enabled site vs. casting a tab]. Cast-enabled computing devices installed with various of Google's own Cast-enabled apps are programmed to perform this functionality, including but not limited to the YouTube, YouTube Music, YouTube TV, and YouTube Kids apps, as illustrated by the following screenshots:  Playing on [redacted] Office TV Kevin Stefanski: "We gotta pour everything we have into game one" Cleveland Browns SUBSCRIBED Share Create Download Comments 110 Save up to $800 on Samsung phones $399 Next: Jarvis Landry: "We know who we are" TV queue • 1/1 IMAGINE DRAGONS EVOLVE Believer Imagine Dragons UP NEXT LYRICS RELATED